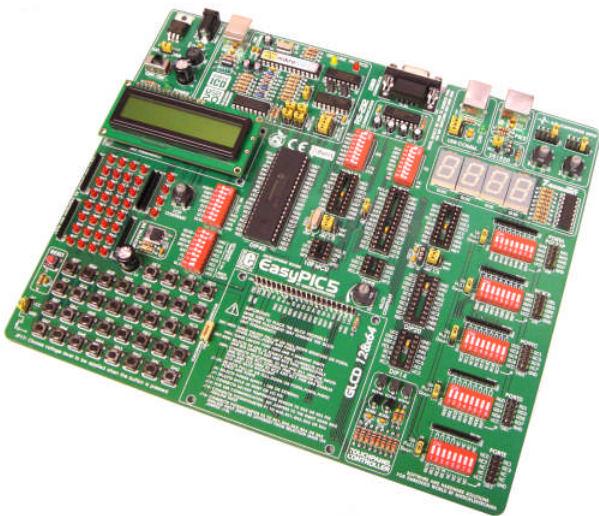
型番:PICD-500EX5　　REV.3.60

USB2.0対応 PICマイコンライター搭載

PICマイコン開発用統合評価ボード 5th Version

取扱説明書

お使いになる前にこの説明書をよくお読みの上正しくお使いください。

ボード本体

製品の概要

PICマイコン開発用統合評価ボード[型番:PICD-500EX5、以下PICD-500EX5と記載]はオンボードでUSB2.0対応のPICプログラマーを搭載し、様々な周辺回路を実装した開発・実験・試作・学習用とあらゆる用途に使用できる統合開発ボードです。

ボード上には、8ピン・14ピン・18ピン・20ピン・28ピン・40ピンのICソケットが用意されており、現在流通しているほぼすべての8ビットPICマイコンを装着できます。どのピン数のデバイスであっても、ジャンパー設定を行うだけでボードに搭載されている各周辺回路を利用することができます。

ボード上に搭載のUSB接続PICライターは、数多くのPICマイコンに対応しており、基板上に装着したデバイスにプログラムを書き込むことができます。USB接続を行った場合、USBバスからPICD-500EX5の電源を取りますので、別途電源を準備する必要がありません。

ボードに搭載されている各周辺回路に接続されているピンは周辺回路に接続して利用するか又は周辺回路には接続しないで使用するかをディップスイッチやジャンパーソケットで選択できます。また全I/Oピンはボード右側に実装されているヘッダピンから取り出すことができます。

コードサイズ限定版(2Kワードまで)の体験版Cコンパイラーを付属。ICD(インサーキットデバッグ)機能搭載なので手軽にC言語によるICD機能が体験できます。また、本書には簡単なC言語チュートリアルを収録していますので、すぐにC言語による開発を体験できます。

パッケージの内容

■同梱物

・PICD-500EX5ボード本体
・PIC16F887（ボードに装着済み）
・8MHz水晶発振子　（ボードに装着済み）
・16文字×2行液晶ディスプレイモジュール　（ボードに装着済み）
・USBケーブル
・拡張用10ピンフラットケーブル
・CD-ROM
・マニュアル(本書)

※PICD-500EX5の全回路図は付属のCD-ROM内データシートフォルダに収録されています。

対応デバイス　（Ver.7.13）

PIC10F200,202,204,206,220,222

PIC12F508,509,510,609,615,629,635,675,683

PIC16F54,57,59,72,73,74,76,77,83,84(A),87,88,505,506,526,610,616,627(A),628(A),630,631,636,639,676,677,648A,676,684,685,687,688,689,690,

PIC16F722,723,724,726,727,737,747,716,747,767,777,785,818,819,870,871,872,873(A),874(A),876(A),877(A)

PIC16F882,883,884,886,887,
PIC16F913,914,916,917,946

PIC18F242,248,252,258,442,448,452,458,1220,1320,2220,2320,2331,2410,2420,2439,2455,2480,2510,2515,2525,2539,2550,2553,2580,2585,2610,2620,2680,2682,2685,4220,4421,4320,4331,4410,4420,4423,4431,4439,4450,4455,4458,4480,4510,4515,4520,4523,4525,4539,4550,4553,4580,4585,4610,4620,4680,4682,4685,6310,6390,6393,6410,6490,6493,6520,6525,6527,6585,6620,6621,6622,6627,6628,6680,6720,6722,6723,8310,8390,8393,8410,8490,8493,8520,8525,8527,8585,8620,8621,8622,8627,8628,8680,8720,8722,8723

※上記のデバイスのうち8～40ピンのDIPタイプのデバイスに対応しています。

ドライバのインストール及びパソコンとの接続

■デバイスドライバーのインストール　(Windows98,ME以外の方)

WIndows2000,XP,Vista,7をご使用のお客様は、PICD-500EX5をパソコンと接続する前に、デバイスドライバーをパソコンにインストールします。

Windows98及びMEをご使用のお客様は本項は読み飛ばして、次の「パソコンと接続する」の項よりお読み下さい。下記の手順で、ドライバーをインストールします。

1　付属のCD-ROMをパソコンのCD-ROMドライブに挿入してエクスプローラー等で内容を表示します。

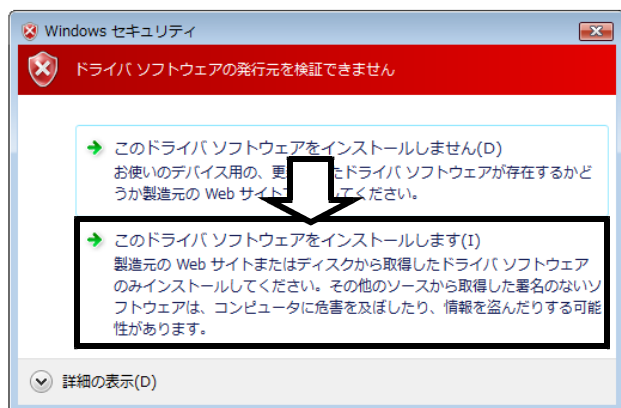
2　"USB Programmer Software"のフォルダを開きます。

3　"Driver"フォルダを開きます。

4　OSごとにフォルダがあります。ご使用のパソコンのOSにあったフォルダを開きます。
※WindowsVista,7に関しては32ビット版用と64ビット版用がありますので間違えないようご注意ください。

5　開くと実行ファイル、"USB18PRG～.exe"というファイルがありますのでダブルクリックして実行します。

6　実行するとダイアログが表示されますので、そのまま"次へ"をクリックして続行します。

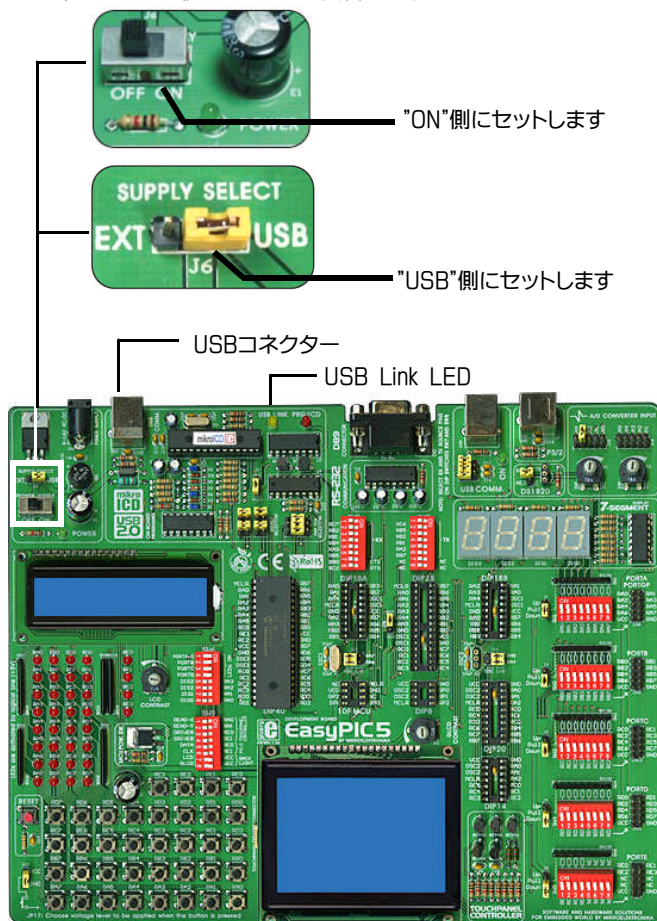
7　ドライバーのインストールが開始されます。
WindowsVista又は7環境で下記のような警告ダイアログが表示された場合には、"このドライバソフトウエアをインストールします"をクリックして続行してください。

※ウイルス対策ソフトウエアなどがインストールされている場合、インストール時にウイルス対策ソフトウエアが警告を表示する場合があります。その場合には、インストールを許可するように設定してください。

8　インストール作業が完了すると、下図のような画面になりますので、"完了"ボタンを押してインストールを完了します。

この時、"状態"のボックスに「使用できます」と表示されていることをご確認下さい。エラーが表示された場合には、ご使用OSの種類と、実行したデバイスドライバーの種類が一致しているかご確認下さい。

■パソコンと接続する

パソコンのUSBポートにPICD-500EX5を接続します。
接続に際しては必ずパソコン本体のUSBポートに接続してください。
USBハブを介しての接続の場合には正しく動作しないことがあります。

1　PICD-500EX5の電源スイッチをONにします。
また、PICD-500EX5の電源をUSBのバスパワーより給電しますので、J6の"USB"側にソケットを装着してください。

※7セグメントLEDの上にあるUSBコネクターは実験用のコネクターですので、通常は使用しません。

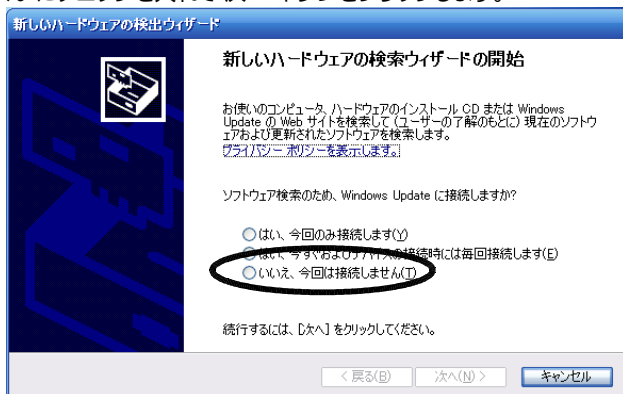
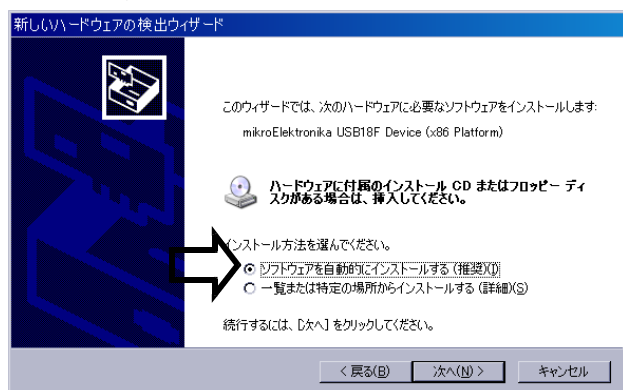
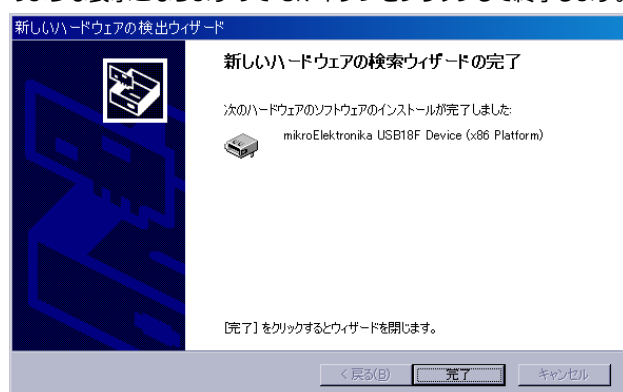
2　USBケーブルで、パソコンのUSBポートと、PICD-500EX5を接続します。

※パソコンのUSBポートがUSB2.0対応の場合には高速書き込みが行えます。USB1.1ポートの場合には通常速度での書き込みとなります。

→接続するとPOWERの緑LEDが点灯します。
→新しいハードウエアの検出ウィザードが自動的に起動します。

以下OSによって操作方法が異なります。

■Windows2000, XPをご利用の場合−−−−−−−−−−−−−−−−−−

3 下図のようなダイアログが表示されたら"いいえ、今回は接続しません"にチェックを入れて"次へ"ボタンをクリックします。

4 下記のようなダイアログが表示されますので、"ソフトウエアを自動的にインストールする(推奨)"にチェックをいれて、"次へ"ボタンをクリックします。

5 「ソフトウエアをインストールしています。お待ち下さい」というメッセージが表示されます。しばらくするとインストールが完了して下図のような表示となりますので"OK"ボタンをクリックして終了します。

■WindowsVista,7をご利用の場合−−−−−−−−−−−−−−−−−−−−

3 WindowsVista,7の場合には、PICD-500EX5を接続すると自動的にデバイスドライバーのセットアップが開始されます。
あらかじめデバイスドライバーは先の手順にてインストールされていますので、WindowsVista,7は自動的にデバイスドライバーを検出してインストールを完了します。

接続後しばらく待ちますと、下図のようなポップアップがタスクバーに表示されます。

上図のホップアップが表示されればインストールは完了です。

■Windows98(SE), MEをご利用の場合−−−−−−−−−−−−−−−−−

3 自動的にデバイスが検出されます。
"新しいハードウエアの追加ウィザード"が表示されたら「次へ」を押して続行します。

4 次のダイアログでは,"使用中のデバイスに最適なドライバを検索する"にチェックを入れて,「次へ」をクリックします。

5 "検索場所の指定"にチェックを入れて,「参照」ボタンを押します。
ドライバの場所を指定するダイアログが表示されますのでCD-ROM内の"USB Programmer Software"フォルダ内の"Driver"フォルダにある"WIN98_ME"フォルダを指定してください。CD-ROMドライブがQドライブの場合にはディレクトリは下記の通りとなります。
Q:¥USB Programmer Softeare¥Driver¥WIN98_ME

指定したら「次へ」を押して続行します。
インストールが完了したら,「完了」ボタンを押して終了します。

〜ここからは各OS共通の項目です〜

PICD-500EX5ボード上の"USB Link"の黄LEDが点灯していることを確認してください。
黄LEDの点灯はPC側にPICD-500EX5が正しく認識されていることを示します。

※"USB Link"のLEDが点灯していない場合には、正しくドライバーがインストールされていません。再度手順を確認の上、ドライバーをインストールし直しておためしください。

※デバイスマネージャーに"USB Devices by mikroElektronika"が追加されます。

WIndowsVista及び7環境でご使用のお客様へ ユーザーアカウント制御無効設定のお願い

WindowsVista及び7環境で、PICD-500EX5をご使用の場合、Windowsに搭載されたユーザーアカウント制御(UAC)機能がUSBポートへのアクセスを拒否することにより、下図のようなエラーメッセージが表示されて、書き込みができないという現象を確認しています。

この現象は、WindowsVista及び7において標準で有効になっているユーザーアカウント制御機能を無効に設定することで回避できます。下記の方法で設定を変更して頂けますようお願い致します。

※なおこのエラーは、WindowsVista及び7以外の環境でお使いのお客様には関係ありません。

■Windows Vistaのユーザーアカウント制御を無効に設定する

1 スタートボタンをクリックして、コントロールパネルを表示します。

2 コントロールパネルから"ユーザーアカウント"をダブルクリックします。"ユーザーアカウント"アイコンがない場合には、ウインドウ左上の"クラシック表示"をクリックして表示を変更します。

3 "ユーザーアカウント制御の有効化または無効化"のリンクをクリックします。

4 "ユーザーアカウント制御(UAC)を使ってコンピューターの保護に役立たせる"のチェックを外します。

5 "OK"ボタンを押して完了します。パソコンを再起動します。

■Windows7のユーザーアカウント制御を無効に設定する

1 スタートボタンをクリックして、コントロールパネルを表示します。

2 コントロールパネルから""ユーザーアカウントと家族のための安全設定"をクリックします。

3 "ユーザーアカウント"をクリックします。

4 "ユーザーアカウント制御設定の変更"をクリックします。

5 画面に表示されるスライドバーを一番下の"通知しない"に設定します。

6 "OK"ボタンを押して完了します。パソコンを再起動します。

ソフトウエアのインストール

PICマイコンの開発に必要なソフトウエアをインストールします。CD-ROMをパソコンに挿入して下記の手順でインストールしてください。

■MPLABのインストール

MPLABはPICマイコンのプログラム開発用の統合開発環境です。PICマイコンの開発元マイクロチップ社から提供されています。MPLABがお手持ちのパソコンにインストールされていない場合には、インストールしてください。

1 CD-ROM内の"MPLAB"フォルダを開きます。
"SETUP.EXE"をダブルクリックして実行します。
"Next>"をクリックして続行します。

2 "License Agreement"のダイアログが表示されたら内容をよくご確認の上、同意する場合には"I accept the terms of the licence agreement"にチェックを入れて、"Next>"クリックして続行します。
なお同意しないとインストールは続行できません。

3 "setup type"の選択ダイアログではすべてのプログラムをインストールしますので"Complete"にチェックを入れて"Next>"をクリックします。

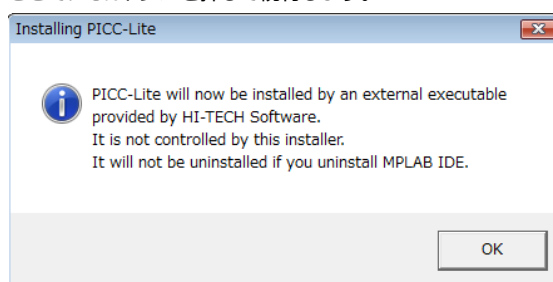
4 MPLABをインストールするディレクトリを指定します。
特に設定する必要がなければデフォルトのままにしておきます。
インストールするディレクトリを変更したい場合には,"Browse"ボタンを押してインストールディレクトリを設定します。
なお変更した場合には,インストールディレクトリをメモするなど忘れないようにしてください。
"Next>"をクリックして続行します。

5 以降のダイアログでは次のように進めます。

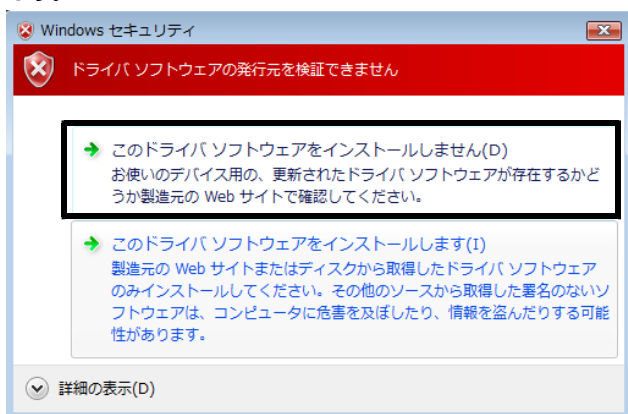
■Application Maestro License
→"I accept the terms of the license agreement"にチェックをいれて"Next>"をクリックします。

■MPLAB C32 License
→"I accept the terms of the license agreement"にチェックをいれて"Next>"をクリックします。

■Start copying files
→"Next>"を押します。
・・・インストールが開始されます。完了までしばらく待ちます。

6 インストールが完了すると、下図のようなダイアログが表示されます。MPLABバージョン8.00以降からはPICCコンパイラーのライトバージョンがMPLABに付属するようになったためです。
ここではOKボタンを押して続行します。

7　下図のようなPICC-Liteのインストール画面が表示されますので、ここでは"Cancel"ボタンを押してインストールをキャンセルします。

※PICC-Liteは、Hi-Tech社のPIC用Cコンパイラーです。MPLAB8.00以降ではライト版(機能限定版)が付属しました。インストールしていただいてもかまいませんが、本キットではこのソフトウエアは使用しません。

8　Windows Vista、7環境にインストールされている場合には、最後に下図のようなドライバーインストールに関するセキュリティ警告が表示されることがあります。
この場合には、"このドライバソフトウエアをインストールしません(D)"をクリックしてください。
ダイアログが消えたら"Finishボタンを押してインストールを完了します。

■2Kワード限定版mikroC PRO 2009 for PIC

CD-ROMには、PIC10F/12F/16F/18Fシリーズの8ビットPICマイコン用、ANSI規格準拠のCコンパイラー、mikroC PRO 2009 for PIC(以下mikroCと記載)が収録されています。
mikroCは、豊富な組込関数を搭載している他、分かりやすいインターフェイスで初心者の方でもすぐになじめるPIC用のCコンパイラーです。

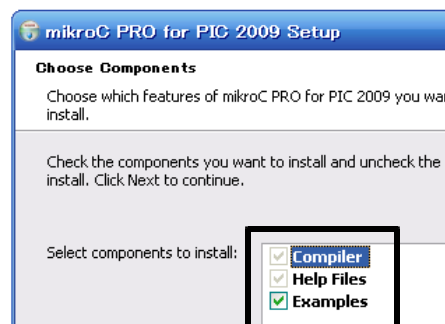
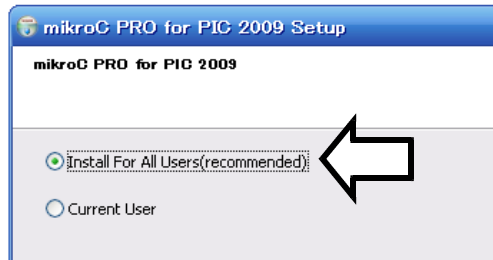
本PICD-500EX5には、生成するHEXファイルのコードが2Kワードまでにサイズ制限されている体験版が収録されています。体験版と言えども、コードサイズに制限がある以外は製品版と同等の機能が利用できます。
使用してみて気に入った場合には、コードサイズに制限のない製品版を当方からPICD-500EX5お買上のお客様特別価格にてお買い求め頂けます。詳しくは別紙のご案内をご覧ください。

なお、Cコンパイラーを使用しない場合には、インストールしなくてもかまいませんが、インストールしない場合には、次の「PICプログラマー用ソフトウエア」を必ずインストールしてください。

1　CD-ROM内の"Cコンパイラー"フォルダを開きます。
"SETUP.EXE"をダブルクリックして実行します。
"Next>"をクリックして続行します。

2　"License Agreement"のダイアログが表示されたら内容をよくご確認の上、同意する場合には"I accept the terms in the License Agreement"にチェックを入れて、"Next>"クリックして続行します。
なお同意しないとインストールは続行できません。

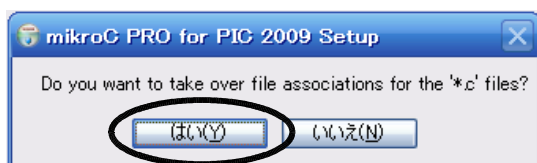
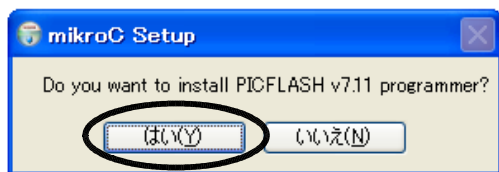
3　次の画面では、"Install For All Users(recommended")にチェックを入れて、"Next>"ボタンを押して続行します。

4　インストールするソフトウエアコンポーネントを選択します。下図の通り、3つすべての項目にチェックが入っていることを確認します。

(CompilerとHelp Filesはグレーアウトしていることがありますが、そのまま続行してください。)
"Next>"ボタンを押してください。

5　インストールディレクトリを選択します。
通常はそのままの設定でかまいませんが、変更する場合には、"Browse"ボタンをクリックして、インストール場所を指定してください。
なお、インストール場所にサンプルプログラムなどもインストールされますので、インストールした場所はメモを取るなどして覚えておいてください。

※ディレクトリ名に日本語や2バイト文字が含まれないようにしてください。

設定が完了したら、"Install"ボタンをクリックします。
→インストールが開始されます。

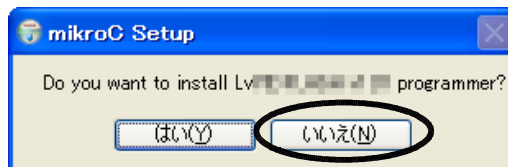
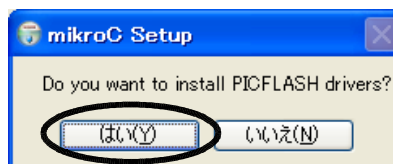
6　インストール途中で、右図のようなメッセージが表示されることがありますので、その場合には"はい"をクリックして続行します。

7　インストールが完了すると、"Finish"ボタンが表示されますので、クリックしてインストールを完了します。
引き続き、下図のようなメッセージが表示されますので、"はい"をクリックします。

8　書き込みソフトウエアをインストールします。インストールウィザードが起動しますので、"Next>"をクリックして続行します。

→ライセンス許諾画面が表示されますので、内容をお読みになって、同意される場合には、"I accept the terms in the License Agreement"にチェックを入れて、"Next>"をクリックします。

※同意しないとインストールは続行できません。

9　インストールするソフトウエアを選択します。
ここでは"PICFLASH"にチェックが入っていることを確認して、"Next"をクリックします。
続いて、インストールするディレクトリを選択するダイアログが表示されます。通常はそのままの設定でかまいませんが、インストールするディレクトリを変更場合には、"Browse"ボタンを押して、インストール場所を変更してください。

インストール場所を設定したら、"Install"ボタンを押してインストールを続行します。

10　インストールが完了したら"Finish"ボタンを押して、インストールを終了します。
引き続いて、次の図のようなメッセージが表示されますので、"いいえ"を選択します。

さらに"Do you want to install PICFLASH drivers?"というメッセージが表示されますので、"はい"をクリックします。

11　これで、ソフトウエアのインストールは完了です。

■PICプログラマ用のソフトウエア

PICへプログラムを書き込む際に使用するプログラマソフトウエアのインストールを行います。mikroCコンパイラーをインストールしなかった場合のみインストールしてください。mirkoCコンパイラーをインストールされた場合には、一緒に書き込みソフトウエアもインストールされますので、この項目は読み飛ばしてください。

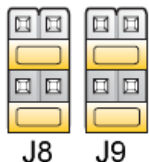
【注意!　必ずお読み下さい　】
mikroCコンパイラーをご使用になる場合には、mikroCをインストールして頂き、ここで説明しているソフトウエアはインストールしないでください。ソフトウエアが二重にインストールされてしまい、不具合の原因になる場合があります。

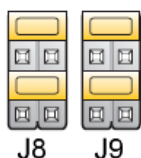
1　CD-ROM内の"USB Programmer Software"内にある"SETUP.EXE"をダブルクリックして実行します。インストーラーが起動しますので、画面の指示に従ってインストールしてください。

2　"インストールは完了しました"ダイアログが表示されたら,「完了」をクリックして終了します。

使用するデバイスのピン数に応じた設定

PICD-500EX5では本体のソケットに装着するPICのピン数に応じてジャンパーピンJ8及びJ9にて設定を行う必要があります。
使用するデバイスのピン数に応じて、使用前に必ず下記の通り設定を行ってください。

●20、14、8ピンのデバイスを使用する場合

20ピン、14ピン、8ピンのデバイスを使用する場合にはJ8及びJ9のピンを上図の様にジャンパー設定してください。

●40、28、18ピンのデバイスを使用する場合

40ピン、28ピン、18ピンのデバイスを使用する場合にはJ8及びJ9のピンを上図の様にジャンパー設定してください。

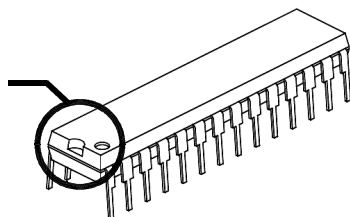
※本体に最初から装着されているPIC16F887は、40ピンデバイスですので、上記の40ピンデバイスの設定にしてお使いください。(工場出荷時にすでに正しい位置に設定されています。)

PICマイコンの装着

使用するPICマイコンを、PICD-500EX5ボード上のICソケットに装着します。工場出荷時は、PIC16F887がすでに装着されています。

装着できるPICマイコンは、必ず1種類だけです。複数のソケットにPICマイコンを装着すると故障の原因となります。

装着時には装着方向を間違わないようにICソケットの切り欠き部分とPICマイコンの切り欠き部分が合うように正しく装着してください。

切り欠きの方向が合うように

■8ピンPICマイコンの装着

8ピンのPICマイコンにはPIC10FシリーズとPIC12Fシリーズがあります。PICD-500EX5は両方のデバイスに対応していますが、ソケットが異なります。

PIC10Fシリーズの場合には、"10F MCU"と記載されたソケットに装着します。PIC12Fシリーズの場合には、"DIP8"と記載されたソケットに装着します。

■18ピンPICマイコンの装着

18ピンデバイス用のソケットは、"DIP18A"と"DIP18B"と2種類あります。これは、現在流通している18ピンのPICマイコンでは2種類のピン配置のものがあるためです。例えば18ピンデバイスでもPIC16F628Aと、PIC18F1220ではピンのアサインが違っています。

18ピンデバイスを使用する際には、ご使用になるPICマイコンのデータシートでピン名称と配置をご確認頂き、基板上のシルク印刷と実際のデバイスのピン名称が一致するソケットの方にデバイスを装着してください。

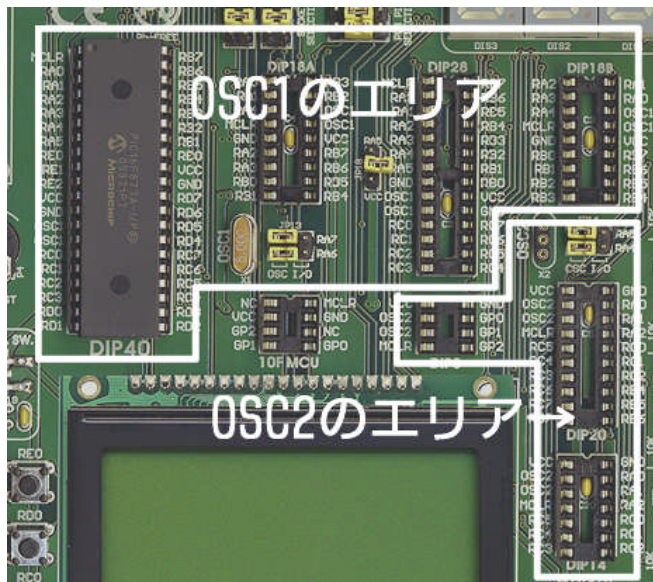
※PICマイコンの1ピンがRA0か、RA2なのかで判断できます。

■PIC18F2331を使用する場合の設定

PIC18F2331をご使用になる場合だけ、基板上J18のジャンパー設定をVCC側にを設定します。その他のデバイスをご使用になる場合には、必ず"RA5"と印刷された側(上側)にジャンパーソケットをセットしてください。

外部発振子の取り付けと、内部/外部発振子の選択

■外部発振子の取り付け

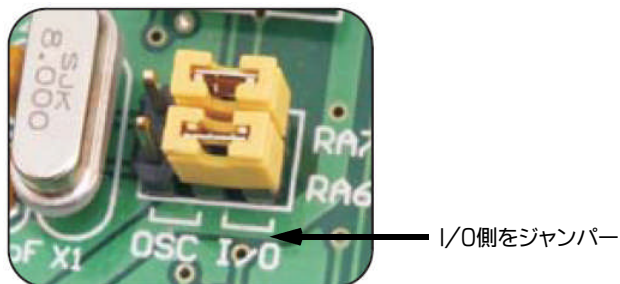
PICD-500EX5では、外部発振子を取り付けることができるよう2つの発振子用ソケットが準備されています。
基板上には、OSC1とOSC2がありOSC1は40ピン・28ピン・18ピンデバイス用のソケット、OSC2は20ピン・14ピン・8ピンデバイス用のソケットです。

外部発振子には、水晶発振子又はセラミック発振子が利用できます。

■内部／外部発振子の選択

PICマイコンの種類によっては、発振子を内蔵しているものがあります。PICD-500EX5では、外部発振子を使用するか、内部発振子を使用するかをジャンパーソケットJ13及びJ14で設定します。
この設定が正しくないとPICマイコンは動作しませんので、使用前によくご確認下さい。

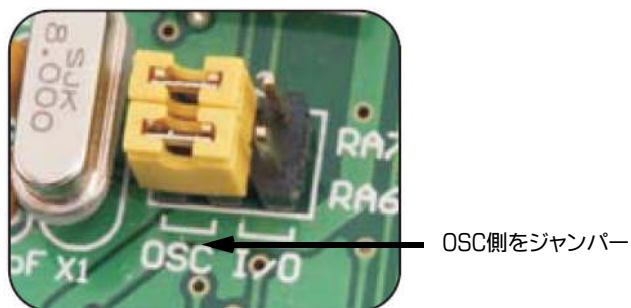
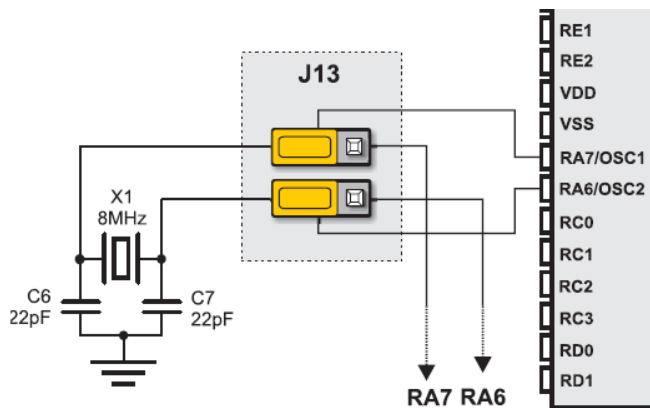
●内部発振子を使用する場合

OSC1のエリアはJ13で、OSC2のエリアはJ14で設定します。
内部発振子を使用する場合には、J13又はJ14のジャンパーソケットを"I/O"と印刷された方(右側)にセットします。
PICマイコンのOSCピンはI/Oポートとして使用できます。(※)

I/O側をジャンパー

※内蔵発振子を使用する場合には、上記のボードの設定の他にプログラム内でOSCCONレジスタなどのオシレーター関連レジスタにおいて内蔵発振子を有効にするような設定が必要です。
また、プログラム書き込み時に設定するコンフィギュレーションビットではOscillatorの種類をINTOSCに設定するなど、各種設定が必要です。

●外部発振子を使用する場合

OSC1のエリアはJ13で、OSC2のエリアはJ14で設定します。
外部発振子を使用する場合には、J13又はJ14のジャンパーソケットを"OSC"と印刷された方(左側)にセットします。

OSC側をジャンパー

内部結線は下図のようになっています。

電源の投入方法

PICD-500EX5は、USB接続にてUSBバスから電源の供給を受けることができます。この場合にはPICD-500EX5には別途電源を供給する必要はありません。
PICD-500EX5側で多くの電流(およそ300mA以上)を消費する予定がある場合にはUSBバスパワー給電では電力が不足する場合があります。この場合には別途外部からACアダプタにて電源を供給する必要があります。
給電方法はジャンパーソケットJ6によって設定を行います。
なお工場出荷時の設定はUSBバスパワー給電の設定になっています。

■USBバスパワーから給電する場合

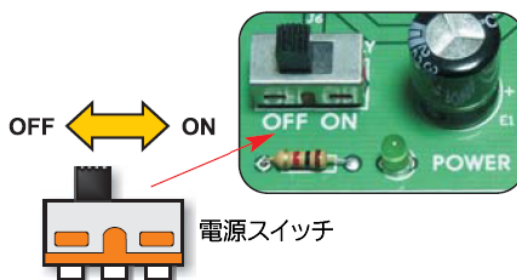
PICD-500EX5ボード上の"SUPPLY SELECT"ジャンパーピン(J6)を「USB」側にショートします。

■ACアダプタから給電する場合

PICD-500EX5ボード上の"SUPPLY SELECT"ジャンパーピン(J6)を「EXT」間にショートします。

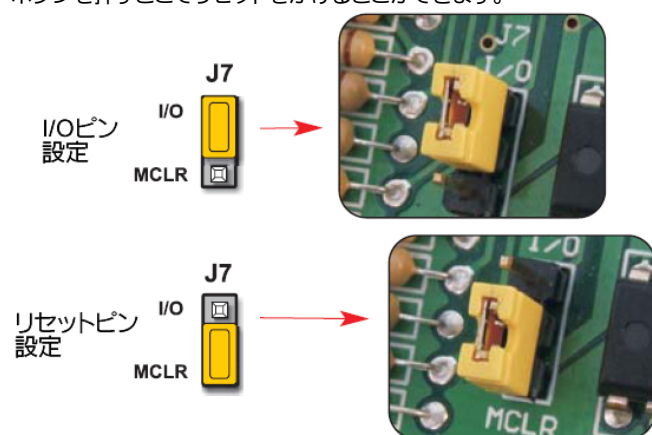
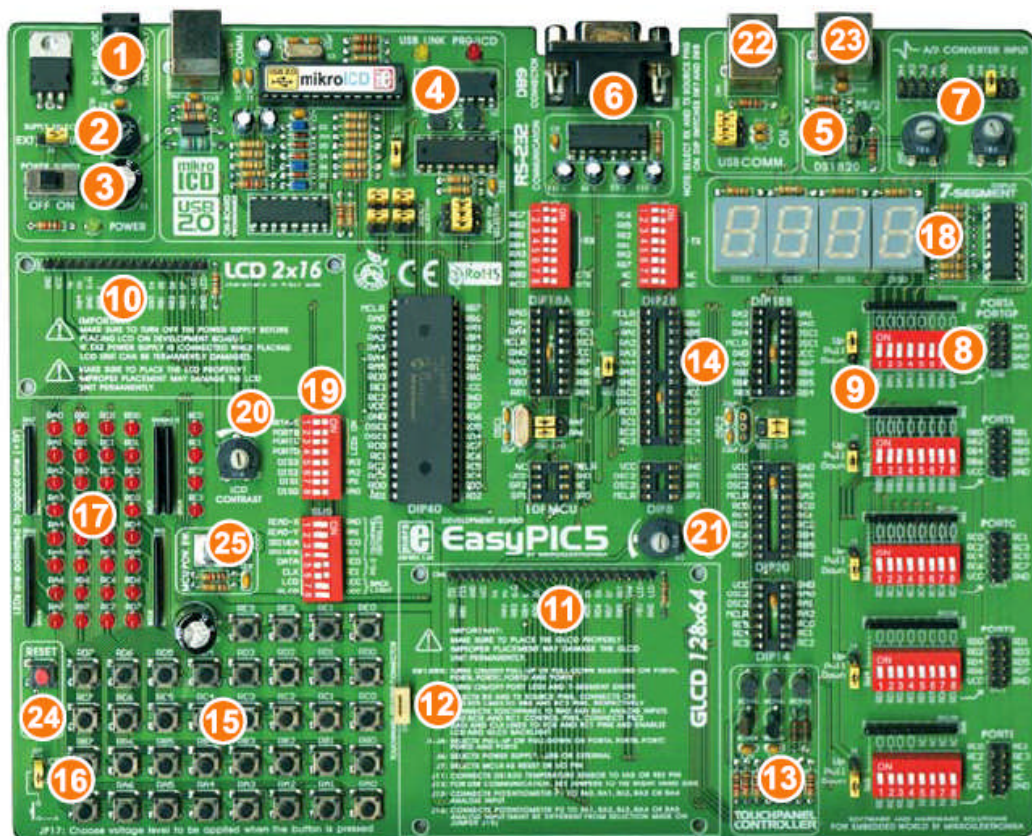
※ACアダプタはDC9V～12Vで500mA以上のものをご用意ください。
径は2.1φです。(センターの極性は問いません)

■電源スイッチ

PICD-500EX5では、電源スイッチが搭載されています。
スイッチによってPICD-500EX5の電源のON/OFFが可能です。

$\overline{\text{MCLR}}$ピンの使用に関する設定

　最近のデバイスでは、$\overline{\text{MCLR}}$ピンをマイコンの外部リセット用のピンとしてではなく、入力ピン（又は入出力ピン）として利用できるデバイスが増えています。
$\overline{\text{MCLR}}$ピンをどちらの設定にするかは、HEXファイルをマイコンに書き込む際に設定するコンフィギュレーションビットで設定を行いますが、ボード上でもジャンパーソケットの設定が必要です。設定はJ7ジャンパーソケットで行います。

J7ソケットを"I/O"と書かれた方に設定すると、$\overline{\text{MCLR}}$ピンは入出力ピンとして利用できます。この設定では基板上のリセットボタンは無効となります。

J7ソケットを"MCLR"と書かれた方に設定すると、$\overline{\text{MCLR}}$ピンはハードウエアリセットのピンとなります。この設定にすると、基板上のリセットボタンを押すことでリセットをかけることができます。

PICD-500EX5の各部の説明

①ACアダプタジャック

外部からACアダプタでPICD-500EX5の電源を給電する場合にはこのジャックにACアダプタを接続します。

DC9V出力、径2.1φで500mA以上が取り出せるものをご使用下さい。(センター極性は問いません)

※給電方法については、②のジャンパー設定が必要です。

②給電方法選択ジャンパー

PICD-500EX5への給電方法を選択します。USBバスパワーから給電するのか、ACアダプタより給電するのかをジャンパーピンで設定します。詳しくは本書7ページの「電源の投入方法」の項をご覧下さい。

外部電源を使用する場合

USBから電源を給電する場合

③電源スイッチ

PICD-500EX5の主電源のON/OFFをすることができます。

詳しくは本書7ページの「電源の投入方法」の項をご覧下さい。

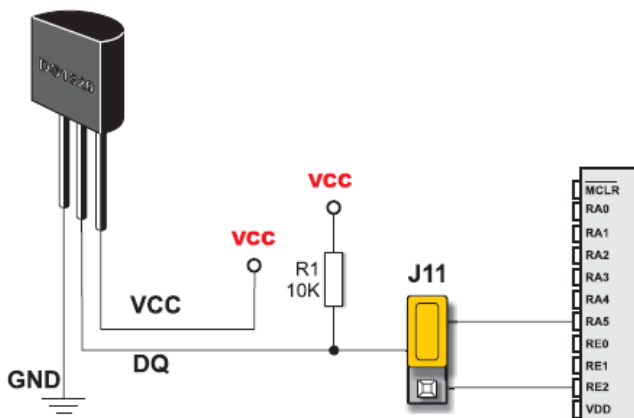
④USB2.0対応オンボードPICマイコンライター部

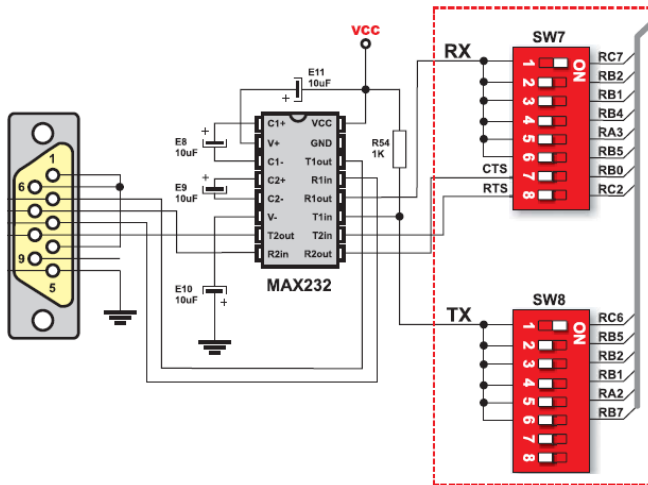
USB2.0に対応したPICマイコンライターです。ボード上に装着した各種PICマイコンにプログラムを書き込みます。

また、CD-ROMに収録のmikroC、Cコンパイラー(限定版)と組み合わせることで、ICD機能(インサーキットデバッグ)が使用できます。

⑤DS18S20、半導体温度センサIC装着用ソケット

オプションの半導体温度センサIC、DS18S20を装着するためのソケットです。

J11のジャンパーによって、DS1820のDQピンをRA5又はRE2に接続できます。なお、DQピンは、10kΩの抵抗でプルアップされています。使用しない場合には、ジャンパーソケットは外しておきます。

⑥RS232Cポート

レベル変換IC搭載のRS232Cポートです。
RX(受信データ)及びTX(送信データ)の信号線の接続先は、SW7及びSW8のディップスイッチによって、選択することができます。
また、CTS線及びRTS線はSW7によってRB0及びRC2へ接続するかどうかを選択できます。

SW7は、RX線の選択用スイッチで、RC7,RB2,RB1,RB4,RA3,RB5から1つを選択します。また、CTS線をRB0に、RTS線をRC2に接続するかどうかを設定できます。

SW8は、TX線の選択用スイッチで、RC6,RB5,RB2,RB1,RA2,RB7から1つを選択します。

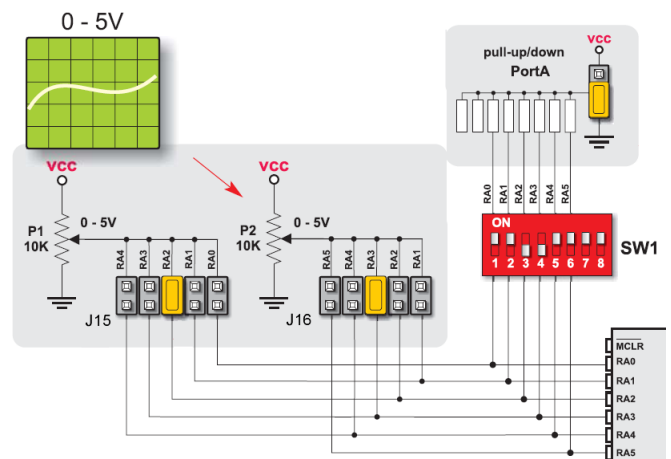
SW7及びSW8のRX線及びTX線はいずれも必ず1つのピンと接続してください。接続するピンについてはプログラムの内容や使用するPICマイコンの種類によって異なります。ハードウエアUART機能を使用する場合には、PICマイコンのデータシートをご覧になり適切に設定を行ってください。

CTS及びRTSは、ハンドシェイク通信を行う場合必要となります。通常ハンドシェイク通信を行わない場合には、OFF側にセットしておきます。

⑦A/Dコンバーター評価用ボリューム

A/Dコンバーター評価用のボリュームです。J15及びJ16のジャンパー設定により、指定したピンに0V～5Vまでの電圧を印加することができます。
ボリュームは2系統あり、J15及びJ16のジャンパー設定によって接続するピンを指定します。RA0～RA5が選択できます。

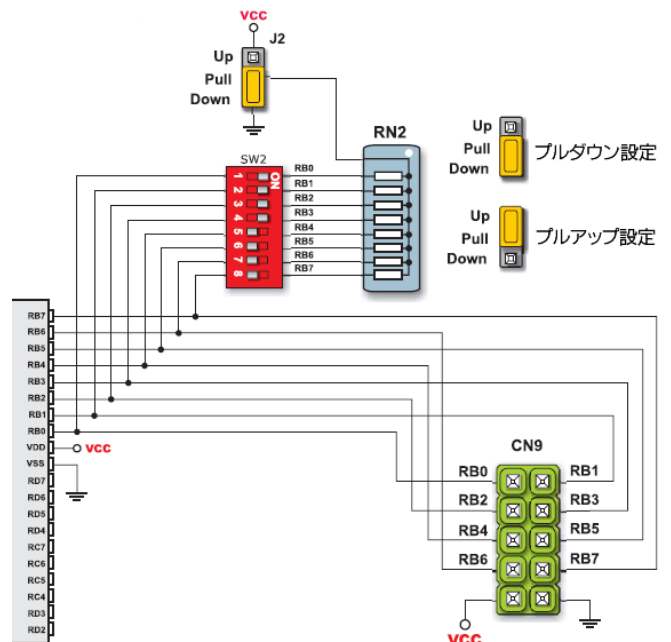
A/Dコンバーターを使用する場合には、アナログ入力として使用するPORTAのピンのプルアップ/ダウン設定をディップスイッチSW1でOFFに設定してください。SW1では、PORTA全ビットを個別にプルアップ/ダウン抵抗と接続するかどうかを指定できますので、J15及びJ16でアナログ入力として指定したピンは、SW1によりOFFに設定します。

内部の結線状態は下図のようになっています。

⑧PORTA～PORTE全ビット取り出し可能ヘッダピン

⑨各ビット個別プルアップ/プルダウン有効・無効設定スイッチ

全I/Oピンをこの部分のヘッダピンから取り出せます。

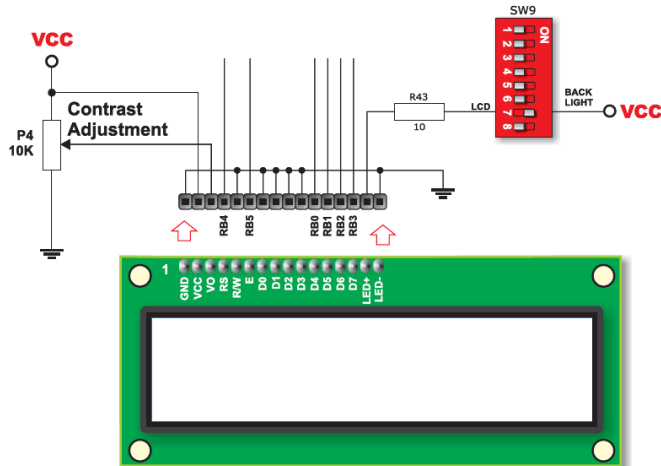
ポート単位でプルアップ/プルダウンがJ1～J5のジャンパーによって設定できます。プルアップに設定すると、10KΩの抵抗器によって、Vccへプルアップされます。プルダウンに設定すると、同様に10KΩの抵抗器によってGNDへプルダウンされます。

また、J1～J5のフルアップ/プルダウン設定は各ポートにおいて、各ビットごとにディップスイッチによって個別に有効にするか、無効にするかの設定ができます。配線図は下記の通りです。

上図はPORTBの配線内容です。
ADコンバーターを使用する場合には、使用するビットはディップスイッチによって、プルアップ/プルダウン設定をOFFに設定します。

⑩2行16桁液晶ディスプレイユニット(4ビットモード)

2行16桁のLCDです。データ線にD7～D4の4ビットを使用する4ビットモードで使用できます。コントラスト調節は、P4ボリュームで行うことができます。

LCDのデータピンD4～D7はそれぞれPICマイコンのRB0～RB3に、EnableビットはRB5に、RSピンはRB4に接続されています。

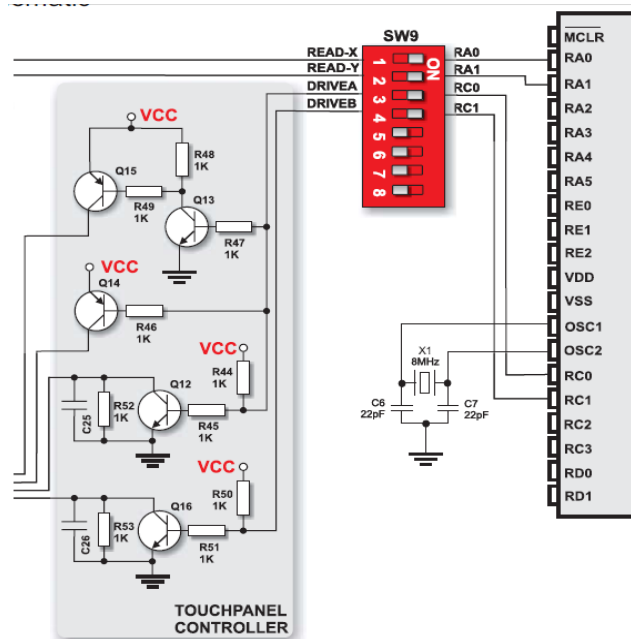
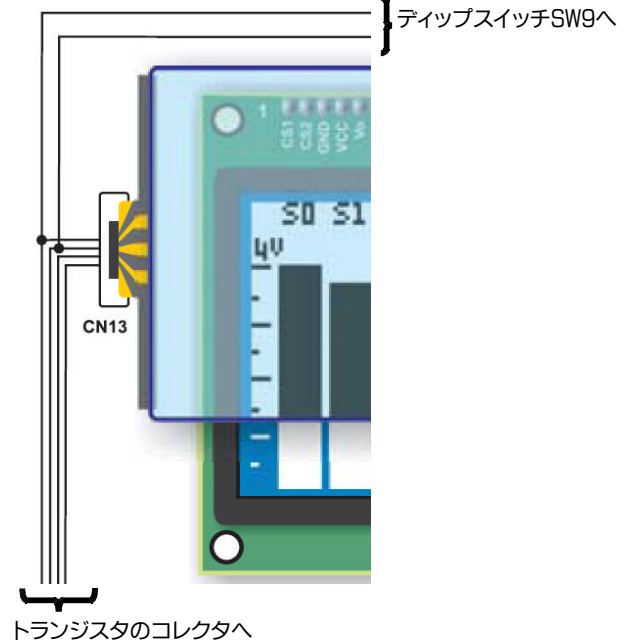
また、バックライト付きのLCDの場合にはディップスイッチSW9の7番スイッチにて、ON/OFFの設定が可能です。

⑪グラフィックLCD用ピン

128×64ドットのドットマトリックスLCDを接続するためのソケットです。

LCDのコントラストは、P3のボリュームで調整します。また、バックライト付きのLCDの場合にはディップスイッチSW9の8番スイッチにて、ON/OFFの設定が可能です。

RSピンはRB2、R/WピンはRB3、EnableピンはRB4にそれぞれ接続されています。また、データピンのD0～D7はRD0～RD7にそれぞれ接続されています。RSTピンはRB5に接続されています。

⑫4線式抵抗膜方式タッチパネル接続用コネクタ
⑬4線式抵抗膜方式タッチパネル駆動用回路

PICD-500EX5には、4線式の抵抗膜方式タッチパネルが装着できます。⑫はタッチパネルのフィルム基板コネクタを挿入するためのソケットです。⑬は4線式抵抗膜方式タッチパネルを駆動するための回路です。⑬の駆動回路は下記のような配線になっています。

駆動回路のトランジスタのコレクターから伸びている配線は、⑫のタッチパネル用コネクタの4線に接続されています。

ドライブAとドライブBは、SW9を介してRC0とRC1へ接続されます。また、READ-XとREAD-YはSW9を介してRA0及びRA1へ接続されます。

タッチパネルの使い方については、C言語でのサンプルプログラムがCD-ROMのサンプルプログラムフォルダに収録されておりますので、そちらをご参照ください。

⑭8ピン～40ピンPICマイコン装着用ICソケット

8ピン～40ピンまでのDIP形状のPICマイコンが装着できるICソケットです。ソケットには複数のPICマイコンは装着できません。必ず1種類だけを装着して使用します。

PICマイコンの装着方法及び外部発振子、内蔵発振子の設定方法等についての詳細は、本書6ページ～7ページに記載されておりますのでそちらをご覧下さい。

⑮32個+4個I/Oピン接続タクトスイッチ群
⑯アクティブLow/High設定用ジャンパー(J17)

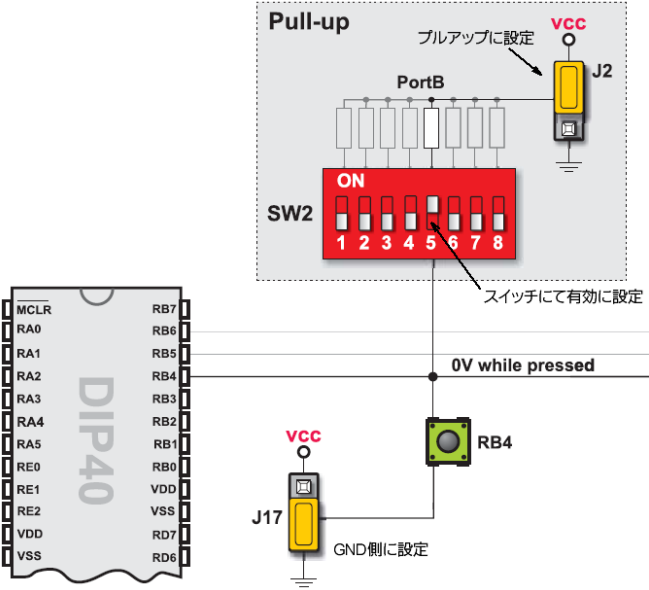
J17のジャンパー設定によって、アクティブHigh又はアクティブLowが選択可能な36個のタクトスイッチです。

タクトスイッチ群の左側にあるJ17をVccと刻印された側に設定すると、アクティブHighに、GNDと刻印された側に設定すると、アクティブLowになります。

※アクティブHighは、ボタンを押した時に該当のピンがHレベルになること、アクティブLowは該当のピンがLレベルになることです。

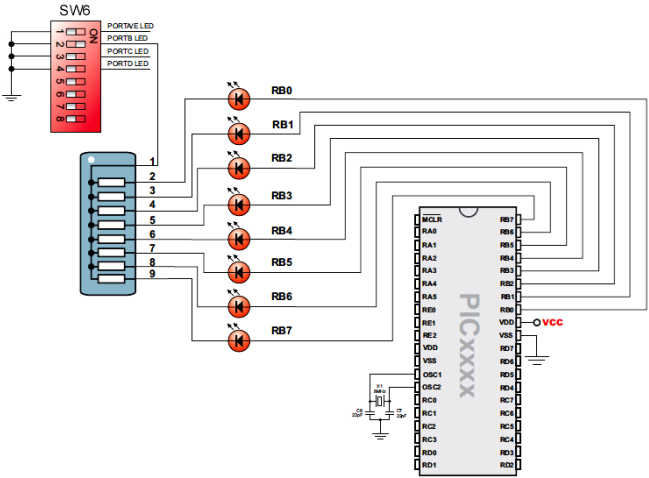
各スイッチを使用する場合には、適切にプルアップ又はプルダウンの設定と、アクティブLow又はアクティブHighの設定を組み合わせる必要があります。各ビットのプルアップ/プルダウンの設定は、⑧及び⑨のジャンパーソケットとディップスイッチによって設定できます。

例えば、RB4のスイッチを使用する場合、ボタンを押していない時はHレベル、ボタンを押した時はLレベルになるように設定したい場合(アクティブLow設定)には、J17のジャンパーをGND側に設定し、続いてJ2にてPORTBをプルアップ設定し、さらにPORTBのディップスイッチのスイッチ5をON側に設定して、プルアップを有効に設定する必要があります。

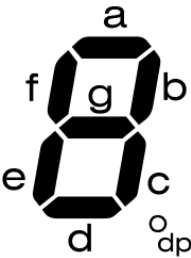
概要図を次に示します。

上図のようにプルアップ/プルダウンの設定と、J17の設定を適切に組み合わせて使用します。

⑰32個+4個全I/Oピン状態表示LED群

PORTA～PORTE（以下RA～REと記載）の全ビットに接続されたLEDです。該当ピンがHighレベルで点灯します。

LEDのカソード側は、基板上のディップスイッチSW6を通してGNDに接続されています。SW6の各ポートのスイッチをON側にするとLEDが使用できるようになります。OFFで物理的に切断されます。

⑱4桁表示7セグメントLED

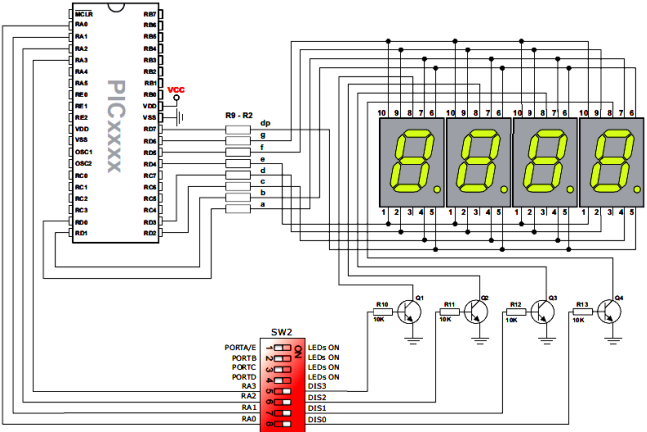
コモンカソードタイプのLEDが4桁搭載されており、ダイナミック点灯方式で数値を表示させることができます。

7セグメントLEDは下図のようにセグメント毎にa～gまでのアルファベットが割り当てられており、各セグメントはPICマイコンのPORTDポートの各ビットに接続されています。

7セグ側	PIC側の結線先
a	PORTD.0
b	PORTD.1
c	PORTD.2
d	PORTD.3
e	PORTD.4
f	PORTD.5
g	PORTD.6
dp	PORTD.7

数字を形成する場合にはPORTDの次のビットをHighレベルにします。

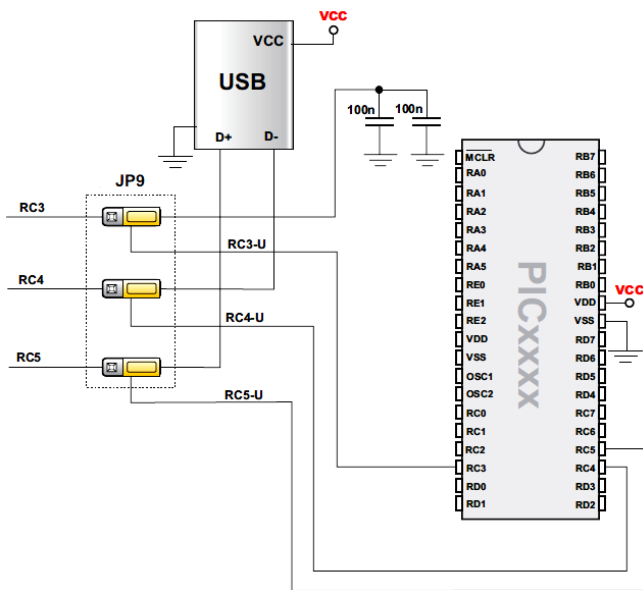
表示数字	Hレベルにするビット	PORTDの値（16進）
0	RD0、RD1、RD2、RD3、RD4、RD5	0x3F
1	RD1、RD2	0x06
2	RD0、RD1、RD2、RD3、RD4、RD6	0x5B
3	RD0、RD1、RD2、RD3、RD6	0x4F
4	RD1、RD2、RD5、RD6	0x66
5	RD0、RD2、RD3、RD5、RD6	0x6D
6	RD0、RD2、RD3、RD4、RD5、RD6	0x7D
7	RD0、RD1、RD2、RD5	0x27
8	RD0、RD1、RD2、RD3、RD4、RD5、RD6	0x7F
9	RD0、RD1、RD2、RD3、RD5、RD6	0x6F

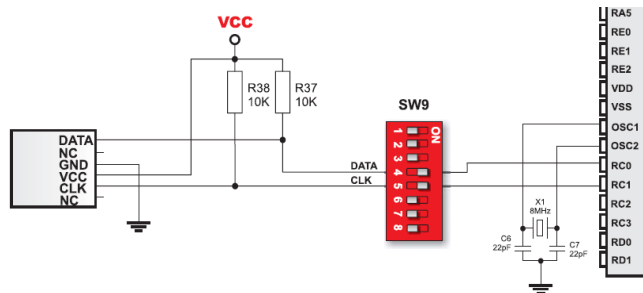
各桁のコモンピンは、トランジスターを通してRA0～RA3に接続されています。該当のビットをHレベルにすることで各桁の表示が有効になります。DIS0はRA0、DIS1はRA1・・・と対応しています。

RA0～RA3を、コモン駆動用のトランジスターと物理的に接続するか、接続しないかの設定は、SW6のディップスイッチにて設定できます。SW6のDIS0～DIS3と印刷されたスイッチをON側にすると、PICマイコンのピンとトランジスターが接続されます。

※7セグメントLEDを使用する場合には、PORTAのプルアップ/プルダウン設定用のディップスイッチのRA0～RA3をOFFに設定してください。

※18ピン・28ピンのデバイスで7セグメントLEDを使用する場合には本書12ページをご覧下さい。

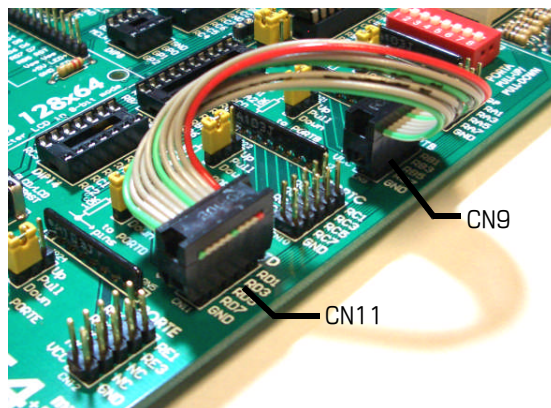
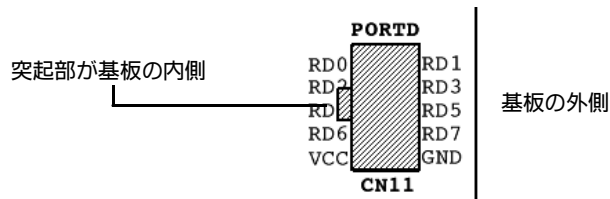
㉒実験用USBポート

PIC18F4550やPIC18F2550などのUSB通信機能搭載デバイスでUSB接続実験を行う際に使用できるUSBポートです。
USB搭載デバイスでUSB通信を行う場合には、J12のソケット3つ全てを右側にセットします。USB通信機能を使用しない場合には、3つ全てを左側にセットします。

㉓PS/2ポート

キーボードやマウスなどでよく使用されるPS/2ポートの実験をするためのポートです。
PS/2のCLKピンはRC1に、DATAピンはRC0に接続されています。両ピンは10kΩの抵抗によりプルアップされています。
接続を有効にするには、SW9の5番、6番にて設定します。使用しない場合には、OFF側に設定します。ON側に設定することで、物理的に配線が接続されます。
なお、PS/2デバイスにはPICD-500EX5側から電源電圧+5Vを供給できます。

㉔リセットスイッチ

PICマイコンのMCLRピンと接続されており、J7のジャンパーソケットが、MCLR側に設定されている時、スイッチを押すとPICマイコンにハードウエアリセットがかかります。
MCLR機能を無効にしていたり、J7ジャンパーがI/O側に設定されている時には、リセットスイッチは動作しません。

18ピン・28ピンデバイスにて 7セグメントLEDを使用する設定について

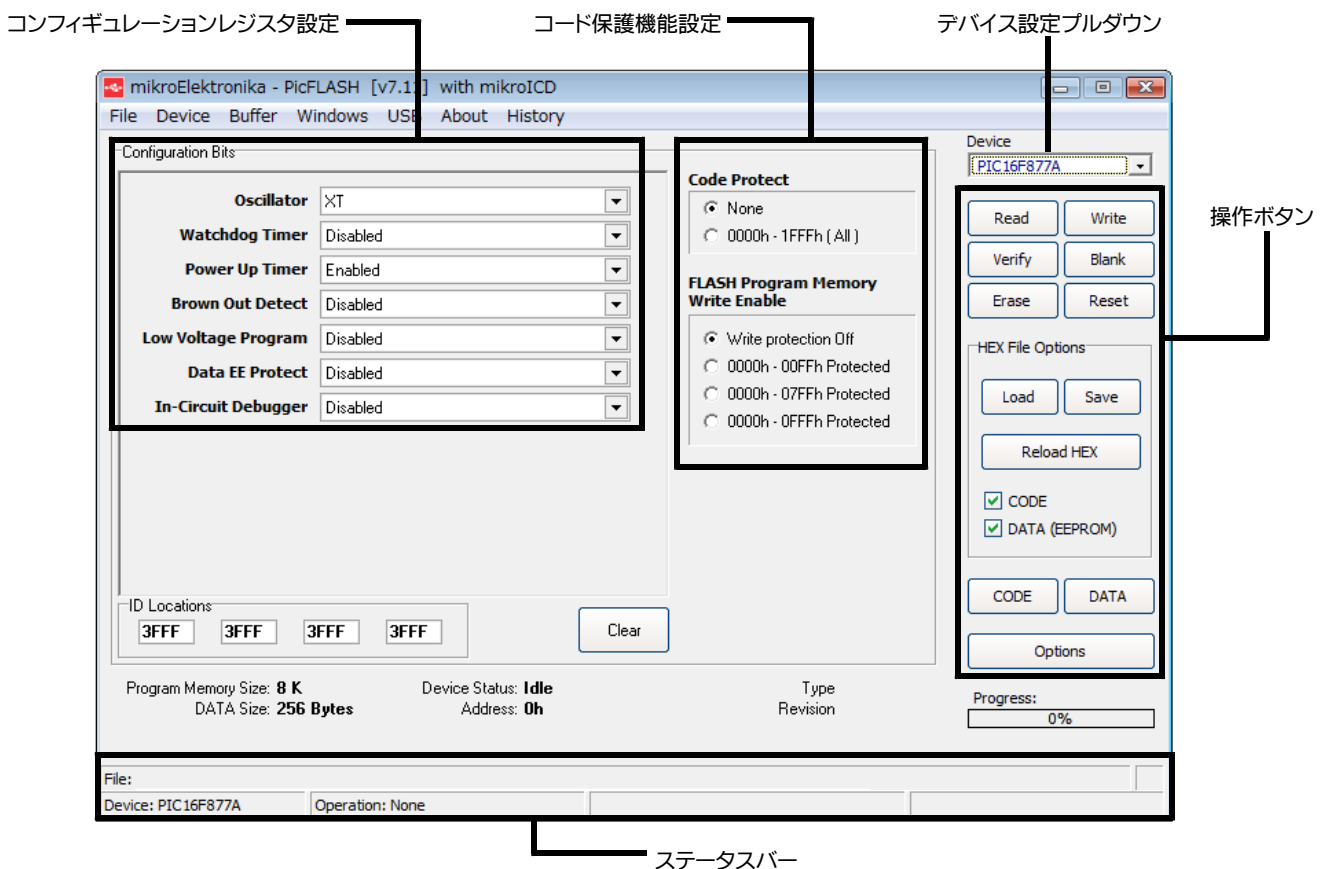
PICD-500EX5では、配線や搭載機能の干渉を避けるため、7セグメントLEDの各セグメントピンは、PORTDに接続されています。そのためPORTDのない18ピンや28ピンデバイスの場合そのままでは使用できません。
18ピン、28ピンのデバイスで7セグメントLEDを使用する場合には、付属の10ピンフラット拡張ケーブルを使用します。
下図のように10ピンフラット拡張ケーブルを、PICD-500EX5基板上のCN9とCN11間に装着します。この接続によりPORTDとPORTBが短絡し、PORTBへアクセスすることで7セグメントLEDが使用できるようになります。

ケーブル装着の際、コネクタの突起がある方が、CN9・CN11両方とも基板の内側に向くように取り付けます。

正しく配線すると、フラットケーブルがSの形に折れ曲がります。取り付け方向を間違えると、機器に損傷を与える恐れがありますで接続には十分ご注意下さい。
この接続により、PORTBとPORTDは短絡されましたので、PORTBへアクセスすることで、7セグメントLEDが使用できます。

なお、この配線は18ピン・28ピンデバイスにて7セグメントLEDを使用する場合にのみ行ってください。7セグメントLEDを使用しない場合にはケーブルは取り外しておきます。また、18ピン・28ピン以外のデバイスを使用する際には、このケーブルは必ず外した状態でご使用下さい。

書き込みソフトウエアの使用方法

付属のPICプログラマーの使い方を紹介します。
PICプログラマーは、MPLAB等で作成したHEXファイルをPICマイコンへ書き込む際に使用するソフトウエアです。

■PICプログラマーの起動
PICプログラマーは、Windowsのスタートメニュー又はデスクトップに作成されたショートカットから起動できます。
スタートメニューから起動する場合には、
"スタート"→"プログラム"→"Mikroelektronika"→"PICFlash with mikro ICD"→"PICFlash"の順でクリックして起動します。

■画面の概要

■操作の順番
次の順番でHEXファイルを書き込みます。

1 デバイス設定プルダウンより、書き込むデバイスを選択します。

2 書き込むHEXファイルを読み込みます。
操作ボタンの"Load HEX"ボタンを押します。ダイアログが表示されますので、ファイルを選択します。

→ファイルが読み込まれるとステータスバーに、ファイル名が表示されます。

※同じファイルを再度ロードする場合には、いちいち上記のようにファイルを指定しなくても、"Reload HEX"ボタンを押すことで再読込ができます。同じファイルを何度もデバッグして書き込む場合などに便利です。

※ファイル名や、ファイルの配置してあるディレクトリ名に日本語や2バイト文字が含まれている場合、正しくHEXファイルの読み込みができません。ファイル名及びディレクトリ名は必ず半角英数字になるようご注意ください。

3 続いてコンフィギュレーションレジスタの設定を行います。
コンフィギュレーションレジスタは、プログラムを書き込む時にだけしか設定できないPICマイコン全体の動作や基本的な設定などを行う特殊なレジスタです。
コンフィギュレーションレジスタの内容はデバイスに搭載される機能により変わるため、設定項目もデバイスにより様々です。ここでは基本的な設定項目についてのみ説明します。

■Oscillator
LP・・・低電力水晶 200KHz以下
XT・・・水晶発振子 4MHz以下
HS・・・高周波水晶、セラミック発振子 4MHz～20MHz
RC・・・RC発振(5KΩと20pFの組み合わせで約1MHz)
EC・・・外部発振子よりTTLレベルのクロック注入
H4・・・HS+PLL内部PLLにてクロックアップ(10MHz発振子接続)
INTRC(IN)・・内部発振,外部に発振子を取り付ける必要なし

■Watchdog Timer Enable
ウォッチドッグタイマのOn/Offを設定します。通常はDisabledに設定します。

■PowerUp Timer
デバイスの電源投入時には、クロックが不安定で動作が不安定になります。この機能を有効にすると、電源投入時一定時間リセットをかけ続けることで、電源が安定した後にクロック動作を開始する機能です。通常はEnabledに設定します。

■Brown Out Detect
一時的な電圧降下の時にハードウエアをリセットする機能です。
PowerUp Timerが有効の時のみ使用できます。

■Low Voltage Programming
低電圧書き込み機能を使用するかどうかを設定します。使用しない場合には必ず "Disable" に設定してください。

■MCLR function
デバイスのMCLR機能(ハードウエアリセット)を有効にするか無効にするかを設定する項目です。Enabledに設定すると、MCLRピンをGNDに接続するとPICはハードウエアリセットされます。
Disabledに設定すると、MCLRピンは入力ピンとして利用可能になります。

[注意]水晶発振子等をお使いで4MHzより高い周波数でお使いの場合には必ずOscillatorの種類をHSに設定してください。またセラミック発振子(レゾネータ)をお使いの場合には周波数に関係なくHSに設定してください。設定が正しくない場合PICマイコンは動作しません。

※外部発振子か内蔵発振子を使用するかにより、J13及びJ14の設定が異なりますので、必ず使用前に設定をご確認下さい。

※内蔵発振子(INTOSC)を使用する場合には、デバイスの種類によってはプログラム内でOSCCONレジスタを設定しなければならない場合があります。詳しくは、使用するデバイスのデータシートをご覧下さい。

4 必要に応じてコード保護機能も設定します。
コードプロテクションを有効にすると、一度書き込んだHEXデータは読み出せなくなります。コードを不正にコピーされることを防止できます。通常は、"None"及び"Write protection Off"に設定します。

5 設定が完了したら書き込みを行います。
PICD-500EX5本体の"USB LINK"の黄LEDが点灯していることを確認してから操作ボタンの"Write"ボタンをクリックします。
書き込みを開始します。

■その他の機能
・Read機能
→PICマイコンからデータを読み込みます。
読み込んだデータは、"Code"ボタンで閲覧できます。またメニューバーの"File"→"Save HEX"で保存できます。

・Verify機能
→現在読み込まれているHEXファイルの内容と、PICに書き込まれているプログラムの内容が一致しているか検証します。

・Blankチェック機能
→現在装着されているデバイスのプログラムメモリがブランク(空)かチェックします。

・Erase機能
→現在装着されているデバイスのプログラムメモリの内容を消去します。

よくあるトラブル事例

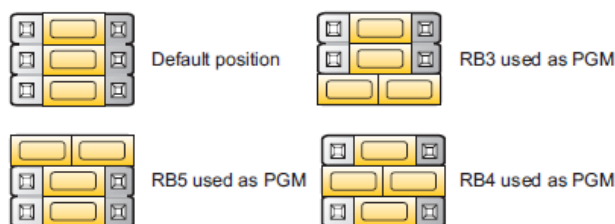
以下にPICD-500EX5を使用する上での注意事項をまとめました。

■RA4のLEDが点灯しない
RA4は、ほとんどのPICマイコンではI/Oピンのドライバーがオープンドレイン方式です。そのため、その他のピンと同様、TTLのように扱おうとしても動作しません。すなわち、このピンはHレベルやLレベルを出力するのではなく、Vssに接続されたFETのオン/オフを出力しています。よってこのピンからHレベルやLレベルをそのまま得ることはできません。

■書き込んだプログラムが動かない
原因は様々ですが、まずは下記の点を再度ご確認下さい。
①Oscillatorの設定は正しいですか?
→5MHz以上の水晶発振子の場合及び周波数に関係なくセラミック発振子の場合にはOscillatorの種類は、HSに設定してください。
②J13及びJ14の設定及びOSC1、OSC2は正しいですか?
③MCLRピンの設定及びJ7の設定は正しいですか?
④J8及びJ9の設定は正しいですか?

■該当ピンをHighにしたのにVcc電圧(+5V)が出力されない
Highレベルに設定したピンがLEDと接続されていませんか?
LEDと接続されていると、LED側に電流が流れるため電圧降下が発生して、ピンの電圧は下がります。SW6でLEDと切り離してお試しください。

■急に書き込みができなくなった、エラーが表示されるようになった
①パソコンを再起動してから、再度お試しください。
②PICD-500EX5のデバイスドライバーを入れ直してください。
③何か外部回路を接続している場合にはすべて取り外しておためしください。

■PIC16F648AやPIC16F876において書き込み時エラーが発生する
PIC16F648Aや一部のデバイスにおいて工場出荷時のデフォルト設定で、低電圧書き込みが有効になっている場合があります。
PICD-500EX5は低電圧書き込みはサポートしておらず、それらのデバイスに書き込みを行おうとするとエラーが発生することがあります。そのような場合には基板上J10のジャンパーソケットを下図のように設定することで、低電圧書き込み用のPGMピンが強制的に設定され、書き込みができるようになります。

ご使用のデバイスのデータシートをご覧頂き、PGMと記載されたピンを確認して、上図の通りジャンパー設定を変更してください。
なお上記の設定にしてプログラムを書き込む際、コンフィギュレーションビットの設定で、Low Voltage Programの項目をDisabledに設定すれば、次回からはJ10はDefault position設定で書き込みができるようになります。

■PICプログラマーでHEXファイルが正しく読み込めない
PICプログラマーでは、長い名前のファイルや階層が深いディレクトリにあるファイルを読み込めない場合があります。
HEXファイルのファイル名は短く、ファイルの保存場所はなるべくルートディレクトリに近い場所に保存してください。またディレクトリ名やファイル名に日本語などの2バイト文字を使用している場合には文字コードの関係上正しく読み込みができません。ディレクトリ名やファイル名には半角英数を使用するようにしてください。

MicroCode Studio(Plus)から連動させて PICD-500EX5のソフトウエアを動作させる設定方法

当方販売中のPIC用BASICコンパイラ、PicBasic Pro Compiler(以下PBPと記載)用の統合開発環境、MicroCode Studio(Plus)から、PICD-500EX5のソフトウエアを連動させて動作させるための設定です。

MicroCode Studio(Plus)(以下、MCSと記載)には、コンパイルとそのコンパイルの結果、できあがったHEXファイルを書き込むためライティングソフトウエアを自動的に立ち上げる機能が搭載されています。
ここで設定を行うことで、MCS上で"Compile and Program"ボタン又は、"ICD Compile and Program"ボタンを押すと、コンパイル後自動的にPICD-500EX5の書き込みソフトウエアが起動します。また、この時コンパイルされたHEXファイルが自動的に読み込まれ、デバイスの型番も自動的に設定されます。PBPをお使いの方は、ここで設定を行うことによってMCS上から連携動作をさせることができるようになります。

1 MCSを起動します。メニューバーの"View"→"Compile and Program Options..."をクリックして開きます。

2 "Programmer"タブへ移動します。

3 "Add New Programmer"をクリックします。

4 "Create a custom programmer entry"にチェックを入れて、"Next"をクリックします。

5 "Display Name"の欄に "PICD-500EX5" と入力して"Next"をクリックします。

6 "Programmer Filename"の欄に "PICFLASH2.exe" と入力して"Next"をクリックします。

7 PICD-500EX5用の書き込みソフトウエアがインストールされているディレクトリを手動で設定します。
"Find Manually"ボタンをクリックします。ダイアログが表示されますのでPICD-500EX5がインストールされているディレクトリを指定します。
デフォルト設定でインストールした場合には通常
C:¥Program Files¥Microtechnica¥PICD-500EX5 Vxxx
となっています。 ※xxxはバージョン番号

8 "Parameters"の欄に、下記の文字列を記述します。文字列には大文字小文字が区別されます。またスペースの位置など間違えないように注意して入力してください。

-pPIC$target-device$ -f"$hex-filename$"
↑
半角スペース1つ

-fのハイフンの前には半角スペースを1つ入力します。

-f の後ろの $hex-filenames$ の部分は必ず " ダブルクォーテーションで囲みます。

9 入力が完了したら"Finish"をクリックして終了します。

主な仕様

電源電圧:	USBバスパワー給電時 DC5V ACアダプタ給電時 DC9V
給電方法:	USBバスパワー又はACアダプタ
USB規格:	Ver,2.0対応
対応OS:	Windows98/ME/2000/XP/VISTA
対応デバイス:	PICマイコン DIPパッケージ
生産国:	セルビア/中国/日本

サポート情報

PICD-500EX5のサポートを行っております。以下のいずれかの方法でご質問をお寄せください。

■FAX番号　03-3700-3548
■電子メール　support@microtechnica.net

ご質問時には、できるだけ詳しくその内容をお書きください。例えば使用しているPICマイコンの型式やパソコンのOS、エラーメッセージが表示される場合には、その内容などをお知らせください。
なお、他社製品に関することや自作回路に関するご質問にはお答え致しかねますのであらかじめご了承ください。

ソフトウエアはアップグレードされることがあります。アップグレードされると対応デバイスが増えたりバグが修正されたりします。
アップグレードの情報などは当方のwebページにてお知らせ致しますので、定期的にご確認ください。

マイクロテクニカ
〒158-0094　東京都世田谷区玉川1-3-10
TEL: 03-3700-3535　FAX: 03-3700-3548

以下はチュートリアルです

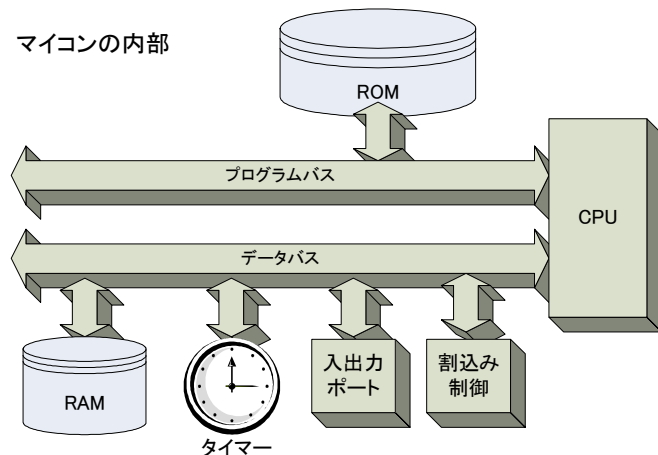
PICマイコンの概要

PICマイコンは、数あるマイコンのうちの1つでアメリカのマイクロチップテクノロジー社が開発しました。"Peripheral Interface Controller"の略で,マイコンの中でも特にワンチップマイコンと呼ばれています。

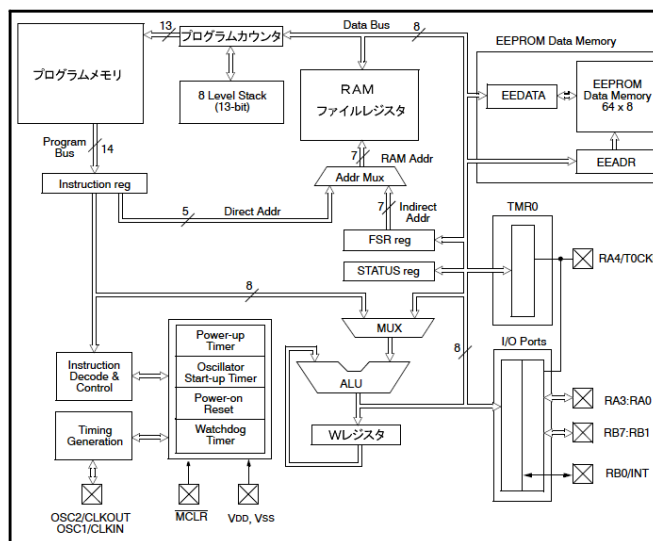
これはその名の通り,ワンチップ,すなわち1つのIC本体の中にほぼすべての機能が詰め込まれていることをいいます。これまでの多くのマイコンは,プログラムを格納するプログラムメモリや,作業用の一時的なメモリスペースであるRAMなどが内蔵されていなかったため,外部に取り付ける必要がありました。しかしPICマイコンはマイコンの動作に必要な機能をワンチップ内にすべて入れてあり,ユーザーインターフェイスであるI/Oポートだけが外部端子に並んでいるので,プログラムが書き込むとすぐに動作を開始するという特徴があります。

PICマイコンには多くの種類があり,今では50種類以上が生産されています。ピン数も8ピン～68ピン以上もある高機能版まで多種多様です。ピン数の違いの方にもそれぞれには少しずつ機能の違いがあり,プログラムメモリの容量が違ったり,A/Dコンバータを搭載していたり・・・・・その選択肢の多さは他のマイコンとは比べものになりません。PICマイコンのアプリケーションを作る際には,まず種類の選定からはじめるのが基本です。

PICマイコンの構造

PICマイコンの基本的な構造すなわちアーキテクチャは,他の一般的なコンピュータのノイマン型アーキテクチャと異なり,"ハーバードアーキテクチャ"という構成を採用しています。
ハーバードアーキテクチャの特徴は,プログラムを格納するメモリ(プログラムメモリ)とデータを格納するメモリ(RAM=ファイルレジスタやデータメモリと呼びます)が別々になっており,それぞれのメモリから,読み出したり書き込んだりするためのデータの通信路であるデータバスが別々になっているという点にあります。

このアーキテクチャのメリットとしては,データバスを別々にすることでプログラムの命令が何ビットの長さになっても,1回でメモリから読み出すことができるため,簡単な構成でかつ高速に命令を実行できる点にあります。他のコンピュータ(ノイマン構造)では、データ長が8ビット単位であることが一般的なため,命令も8ビットの倍数とせざるを得なくなり,1つの命令が複数のバイト構成となるために,メモリからの呼び出し1回だけでは命令全体が読み出すことができなかったり,データとの切替えが都度必要だったと効率が悪くなります。こうなると当然複雑な内部構成となってしまい処理も遅くなってしまいます。
PICマイコンでは,ハーバードアーキテクチャの採用により,12ビット、14ビット・16ビットの3種類のモデルを作り,効率のよい命令実行を行っています。

プログラム大まかな実行の流れとしては,命令がROM(プログラムメモリ)から呼び出され,CPU部分で実行されます。その命令の内容によりデータがRAM(データメモリ)から呼び出されCPUで演算されて結果がまたRAMに格納されたり,I/Oポートにデータが出力されたりします。

実際のマイコンの内部ブロック図を見てみましょう。PIC16F84Aデバイスの内部ブロック図を次に示します。
※PIC16F84AはPICマイコンの中でもシンプルな18ピンデバイスです

●プログラムメモリ(ROM)
プログラムそのものの命令を記憶する場所です。プログラムメモリと呼ばれたり単にROMと呼ばれたりします。ここにプログラムを書き込むためには専用のPICマイコンライターが必要となります。命令実行中にはここの値を書き換えることはできません。

●プログラムカウンタ
プログラムの実行順序を制御するカウンタ。このカウンタの内容が指しているアドレスのプログラムメモリにある命令が次に実行される命令となります。

●ファイルレジスタ(データメモリ)
ファイルレジスタはデータメモリと呼ばれたり単にRAMと呼ばれたりと,名称が混同されていますが,厳密には違いがあります。
ファイルレジスタは2つの領域に別れていて,1つは汎用のデータ格納メモリエリアでRAMと呼ばれ,プログラム内で使う変数領域として使用します。もう一つは特別なレジスタ領域で,一般的に"SFR"(Special Function Register)と呼ばれ,コンピュータ内の動作を決める色々な条件を設定したり、動作状態を知ることができます。
よって,正確にはファイルレジスタは,RAMとSFRから構成されていると理解しましょう。

●Instruction Decode and Control
プログラムカウンタが指すプログラムメモリ内の命令が読み出され,ここで命令種類の解読と処理がなされます。その結果,各種の制御やALU内で演算をしたりすることで命令が実行されます。

●MUX(切り替え部)とALU(算術演算部)
各命令の指示に従って各種レジスタやWレジスタの内容との演算がなされる場所で,いわゆるコンピュータの中の計算器です。演算結果は再度Wレジスタやレジスタに格納されたりI/Oポートやタイマ、プログラムカウンタなどに格納されます。

●Wレジスタ(W reg)
演算をするときに一時保管用に使うレジスタで,演算の時の中心となって働きます。

●入出力ポート(I/Oポート)
内部のデータを外部に出すためのポートで,これがPICマイコンピンに直結されています。
そのためVccが5Vの時は,コンピュータがポートに"1"のデータを出力すると,ICの対応するピンが"High"レベルの状態になり"0"を出力すると"Low"の状態となります。

PICマイコンの種類

PICマイコンには様々な種類があり、選択の幅が広いマイクロコントローラーです。ピン数も8ピンから80ピンのものまであり、用途に応じて使い分けることができます。

PIC10FシリーズはPICの中でも最も小さいマイコンです。DIP形状のものは8ピンからあります。PIC12Fシリーズは、8ピンながら高機能でADコンバータを搭載したモデルもあります。

PIC16Fシリーズは、PICのスタンダードモデルです。14ピン～40ピンまで数多くのデバイスが流通しています。以前はPIC16F84などが主流でしたが最近では高機能なPIC16F88や18ピンの代表格となっています。40ピンデバイスでは、PIC16F887が代表格となっています。

PIC18Fシリーズは、PIC16Fシリーズを高機能化したもので様々な機能が付加されている他、最大の特徴はバンクの概念がなくなったことです。バンクの概念は高級言語を使用している分にはそれほど問題にはなりませんが、アセンブラを使用している場合には、かなりプログラムの仕組みが簡単になりました。
USB2.0機能搭載のデバイスや、ROM容量の大きなデバイス、ピン数が多い80ピンのデバイスなど、PIC16Fシリーズでは事足りない場合の選択肢としてPIC18Fシリーズがあります。

これまでのPIC10F、12F、16F、18Fシリーズはいずれも8ビットMPUです。8ビットマイコンでは、世界シェアNo.1であり市場の18%程度をPICが占めています。
PICも高機能化、高速化の波にのり最近では、16ビットマイコンのデバイスも登場しています。PIC24シリーズは16ビットPICマイコンの代表格です。dsPIC30F、33Fシリーズは16ビットマイコンにDSP機能を追加した特殊なマイコンで、デジタル信号処理を得意としています。
また、最近ではついに32ビットのPICも登場し、PIC32として注目を集めいてます。但し、32ビットマイコンの世界ではすでに多くのメーカーから高機能なCPUが発売されており、PICの特徴を活かせるかは今後の展開によるところです。

マイクロコントローラーの世界では「大は小を兼ねません」。適材適所である必要があります。開発するシステムの規模に応じて、最適なデバイスを選択することで、性能だけではなく消費電力や価格、デバイスサイズなど、色々な問題を最適化することができるためです。PICは豊富なラインナップにより、様々なニーズに応えられるデバイスとして世界的に有名になりました。

PICマイコンの動作概要

プログラムは常に0番地からスタートします。電源投入時やハードウエアリセットがかけられた状態がこれに相当します。
プログラムの進行は,プログラムカウンタによって制御されています。プログラムカウンタの指し示す値が実行される番地を示しています。リセット時プログラムカウンタは0になり,0番地の命令より順番に実行されていきます。1つの命令が読み出されると,演算ユニットALUにて演算や処理が行われます。必要に応じてデータメモリからデータを読み出したり,書き込んだりも行います。
演算ユニットは,プログラム側から見るとWレジスタの形で存在している点に注意が必要です。I/Oポートは,データメモリ上に割り当てられていてメモリ操作によって,アクセスできます。この構造のことを特に"メモリマップドIO"と読んでいます。すなわち,メモリに0や1の値を書き込むことで直接I/OポートがLレベルになったり,Hレベルになったりします。

PICマイコンはRISC(Reduced Instruction Set Computer)タイプのマイコンなので,1命令1クロックで動作します。クロックとは外部から入力されたある一定の周期の矩形波パルスで,通常は水晶発振子やセラミック発振子といった専用の発振子を用いてクロックを生成します。PICマイコンではデバイスによって20MHz～40MHz程度のクロックを扱うことができます。

プログラムメモリから命令を取り出すことを"命令をフェッチする"といいます。PICマイコンでは,1クロック毎に命令はフェッチされています。フェッチされた命令は,インストラクションデーコーダによって解析れさます。なお,PICマイコンは内部で,発振子より与えられたクロック周波数を1/4にして命令を実行しています。すなわち,4MHzのクロックを入力している場合には,内部動作は1MHz動作になっています。

では実際の動作はプログラムを作りながら学習しましょう。

開発言語

PICマイコンの開発言語としては、アセンブラ言語・BASIC言語・C言語などがあります。

アセンブラ言語は、フリーで使用できますが記述内容が直感的でなく習得には時間がかかります。そのかわり、マイコンの内部処理を1つずつ記述していくため、マイコンの仕組みや処理の仕組みが理解できます。日常作業では、高級言語(C言語やBASIC言語など)を使用する技術者でもアセンブラ言語を理解しておくと何かと役に立ちます。

BASIC言語を使用するには、BASICコンパイラーが必要です。当方では、PICマイコン用に開発された、高機能なBASICコンパイラー、PicBasic Pro Compilerを販売しております。直感的に記述できる書式、豊富な組込命令を搭載しており、PICマイコンの機能をフルに使用できます。高度な機能を簡単な記述で実現できます。

C言語を使用するには、Cコンパイラーが必要です。本製品のCD-ROMには2Kワード限定版の体験版Cコンパイラー、mikroC PRO 2009 for PICを収録しており、すぐに体験することができます。mikroCは、ANSI規格に準拠したCコンパイラーで、豊富な組み込み関数を備えており、高度な機能を簡単な記述で実現できます。

本書では、このmikroCを用いてPICの世界を簡単に体験できるよう、以降にチュートリアルを収録しました。是非一度お試し下さい。

PICD-500EX5のチュートリアル　～C言語編～

■C言語とは
C言語は、ANSIという団体によって規格化された高級言語です。高級言語とはアセンブラ言語のように機械語に近いレベルの開発言語ではなく、より人間がわかりやすい言語体系の開発言語を指します。C言語を導入することでより簡単に、合理的にPICマイコンの開発ができるようになります。
C言語では様々なデータ型を扱えるほか、データを指し示すポインタや配列、関数などが使用でき、より高度なプログラムを記述することができます。

本製品に付属のmikroCコンパイラーはANSI規格に準拠したPICマイコン用のC言語です。PIC10F/12F/16F/18Fシリーズの8ビットPICに対応しています。(この他16ビットPIC対応版もあります。)
体験版のため、開発できるプログラムのコードサイズが最大2Kワードに限定されていること以外、すべての機能が使用できます。mikroCが気に入れば、優待価格にてサイズ制限のないフル機能版を入手することができます。詳しくは別紙をご覧ください。

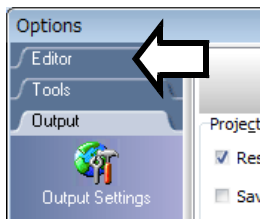
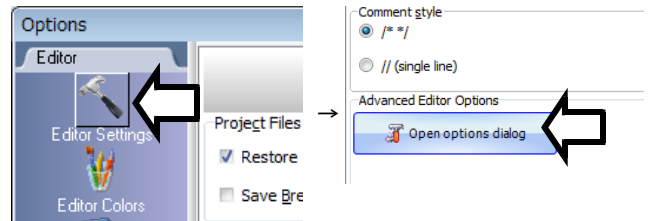
mikroCは、一般的なC言語としての機能の他にPICマイコンの機能をフルに使用できるよう様々な組み込み関数を搭載しています。組み込み関数を使用することで、LCD制御やADコンバーターの操作、UARTなど様々な機能を簡単に実現できます。
また、ICD機能(インサーキットデバッグ)が搭載されており、本PICD-500EX5と組み合わせて使用すると、実機でプログラムを動作させせながらプログラムの挙動を観察したり、1行ずつプログラムを実行させるステップ実行ができるなど、かなりの高機能を実現しています。
一般的にICD機能を行うには別途システムが必要でしたが、本PICD-500EX5とmikroCの組み合わせですぐに実現できます。

C言語チュートリアル　～はじめに～

では、早速付属のmikroC Cコンパイラーを使ってC言語でプログラムを作って実際にPICD-500EX5のボード上で動作させてみましょう。
本マニュアルではC言語の基本から解説することはできませんので、C言語の基本については専門書籍をご参照ください。

■開発環境の設定
では最初にmikroC Cコンパイラーの開発環境を設定しましょう。

1 mikroCを起動します。
"スタート"ボタン→"プログラム"→"Mikroelektronika"→"mikroC PRO for PIC"→"mikroC PRO for PIC"の順でクリックします。

2 起動すると、開発画面が表示されます。
初期設定のままでもかまいませんが、エディタウインドウの文字を大きくしたい場合等は、最初に環境を設定します。
メニューバーの"Tools"→"Options"をクリックします。画面左側のタブから"Editor"をクリックします。

3 メニューが展開しますので、その中から"Editor Settings"をクリックします。その中にある"Open Option dialog"ボタンをクリックします。

4 "Editor Options"ダイアログが表示されますので、画面右下の"Editor Font"部分にある"Font"ボタンをクリックすると、フォントを選択できるダイアログが表示されますので、サイズを調節してください。

※実用的な大きさとしては、画面の解像度にもよりますが、14ポイント程度が見やすいです。

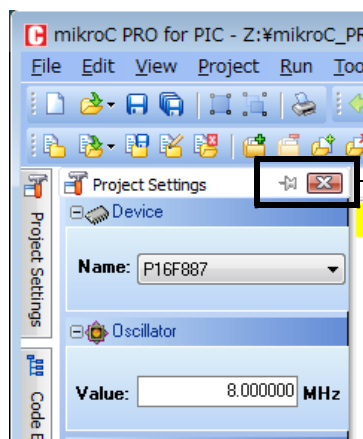
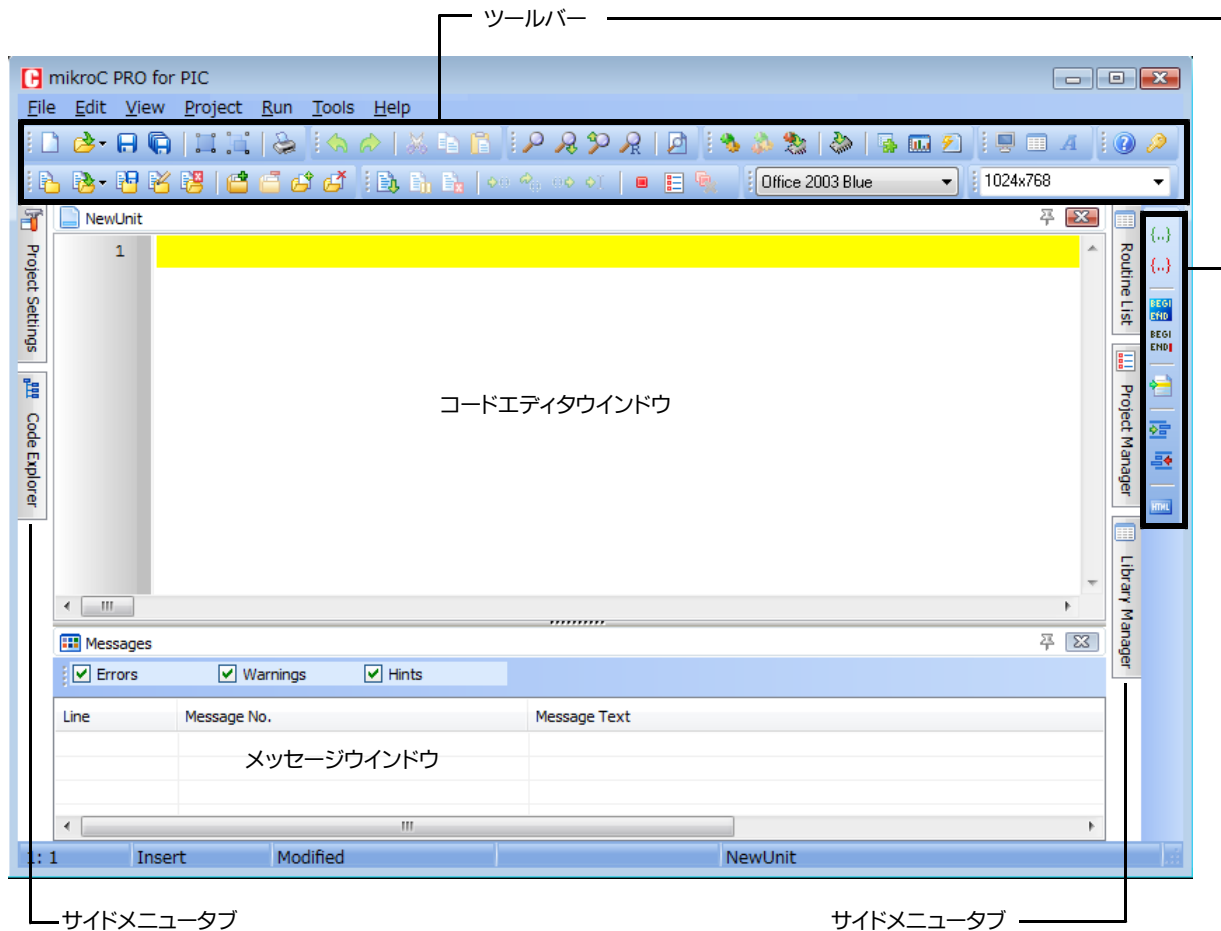
※右上にある"Font"ボタンではありませんので注意してください。

※フォントの種類は変更しないでください。

設定したら"OK"ボタンを押して完了します。さらにOptionsの画面では、"Apply"ボタンをクリックして設定内容を反映させた後、"OK"ボタンを押して完了します。

※その他にも詳細な設定が可能ですが、トラブルを避けるためフォントの変更以外は行わないことをお奨めします。

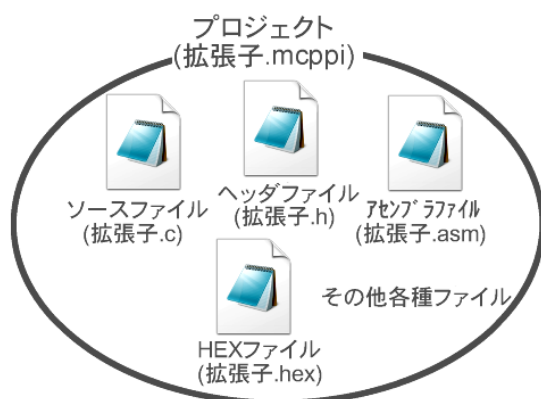
■mikroCの開発環境画面

サイドメニュータブは、便利な機能が割り当てられたタブです。クリックすると該当する画面がサイドから表示されます。デフォルト設定では、マウスのカーソルが上に乗っている時は、メニューは表示されたままとなり、カーソルを外すとメニューは自動的に隠れます。

左図は、"Project Settings"メニューの表示例です。
固定表示されたい場合には、☒ボタンとなりの マークをクリックすると固定されます。また強制的にメニューを隠したい場合は☒ボタンをクリックします。

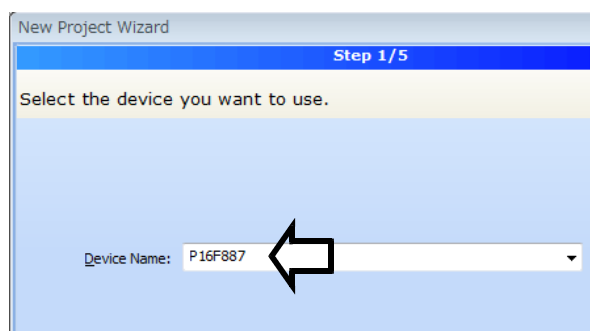
チュートリアル①　～RB0の制御とボタン処理、変数の使い方～

では早速プログラムを作って開発を体験してみましょう。チュートリアル①では次のような内容のプログラムを作ってみます。

int型の変数を用意して、そこに入れた値を2進数でPORTBに出力するプログラムを作ってみましょう。変数を2つ用意して、外部のスイッチ入力によって、どちらの変数の値をRBに出力するのか指定できるようにしてみます。
デバイスは標準で付属しているPIC16F887を、クロックは8MHzを使用します。

【1】プロジェクトの新規作成
プログラムを作り始める前に、プロジェクトを新規に作成します。
mikroCでは1つのファームウエアに対して1つのプロジェクトとして、プロジェクト単位で管理します。
プロジェクトには、これから作成するC言語のソースプログラム(拡張子.c)や、場合によって使用することとなるヘッダーファイル(拡張子.h)などのファイルが登録されます。その他、コンパイル後のHEXファイルなどもプロジェクトに含まれる1ファイルとして管理されます。

よくある間違いとして、プロジェクトの意味を理解していないと画面上に表示されているソースプログラムをコンパイルしているつもりなのに、実際には別のソースプログラムがコンパイルされていて思ったような動作をしなかったり混乱してしまったりというトラブルに繋がります。新しいプログラムを作るときには、必ずプロジェクトを新規に作成するようにしてください。

1　新規にプロジェクトを作成してプログラムを作っていきましょう。
メニューバーから"Project"→"New Project"をクリックします。
mikroCではウィザード形式でプロジェクトを新規に作成します。ウィザード画面が表示されたら、"Next"ボタンを押して続行します。

最初に使用するPICマイコンの型式を選択します。プルダウンから、今回使用する"P16F887"を選択して、"Next"ボタンを押します。

2　続いてクロック周波数を設定します。今回は、PICD-500EX5に最初から付いている8MHzを使用することにします。"Device Clock"の部分に、下図のように "8" と入力して 8MHz に設定します。
小数点以下は自動的に0が挿入されますので、単に 8 とだけ入力してください。

"Next"ボタンを押します。

3　プロジェクトの保存場所と、プロジェクト名を指定します。
ダイアログの右端にあるファイルボタンを押すとディレクトリを指定するダイアログが表示されます。プロジェクトを作成するディレクトリと、プロジェクト名を入力して"保存"ボタンを押します。

ディレクトリ名及びファイル名には、日本語や全角文字などの2バイト文字は使用できません。必ず半角英数字で指定てきるディレクトリ及びファイル名としてください。
デスクトップやマイドキュメントなどは日本語が含まれてしまうことがあり、お奨めできません。

ここでは、例としてCドライブの直下に"mikroC_project"というフォルダを作り、そこに"TEST887.mcppi"というプロジェクトを作成することにします。ディレクトリ名は次のようになります。

C:¥mikroC_project¥TEST887.mcppi

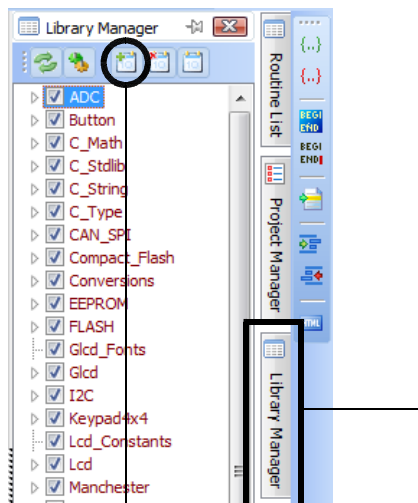
※プロジェクトを作成したフォルダには各種ファイルが作られますので、なるべく分かりやすいディレクトリにすることをお奨めします。

"Next"ボタンを押します。

4　続いて Step4/5 のウィザードでは何もせずに"Next"ボタンを押します。
最後に"Finish"ボタンをクリックして完了します。

【2】ライブラリーの追加
mikroCは、多くのライブラリを用意しています。これらのライブラリを使うことでPICで使うことが想定されるほとんどの機能をプログラムとして簡単に記述できます。
mikroCでは、あらかじめ使用するライブラリを指定することで、実際には使用しないライブラリ関係の冗長部分を少なくしてプログラムサイズを少なくできる特徴があります。しかしながら実際に使用する場合には、どんなライブラリを使っていくかということが見えにくいため、一般的には最初はすべてのライブラリをプロジェクトに登録しておいて、いよいよプログラムが長くなってなるべく冗長部分を削りたい・・・という時だけ使用しないライブラリを外していく・・・という方法をお奨めします。(すべてライブラリを含めていてもそれほどサイズが大きくなるわけではありません。)

以後のチュートリアルでも色々なライブラリを使いますので、ここで紹介する方法で必ずライブラリをプロジェクトに登録してプログラムを開発してください。

1　画面右側のサイドメニュータブの"Library Manager"をクリックして展開します。

2　すべてのライブラリを選択しますので、"Check All"ボタンをクリックします。

3　すべての項目にチェックが入ったことを確認します。

※ここでのチェックを入れ忘れて、該当の組込関数をプログラムの中で記述すると"Undeclared identifier"エラーとなりますので、ご注意下さい。

【3】プログラムの記述

コードエディターウインドウにプログラムを記述しています。
プログラムを書く前にプログラムの概要を下記にまとめます。

符号なし8ビットの変数aとbの2つを用意します。aとbにはあらかじめ値を代入しておきます。
RC0とRC1をアクティブロー(スイッチを押すと該当ピンがLowレベルになるスイッチのこと)のスイッチ入力として使用し、RC0がLレベルになると変数aの値を、RC1がLレベルになると変数bの値を2進数としてPORTB(RB)に出力します。

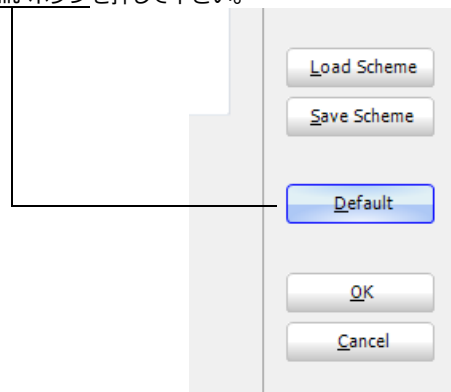
次のように記述しましょう。

```
void main(){

unsigned int a ,b ;

TRISB = 0 ;
TRISC = 0x3;
ANSEL = 0;
ANSELH = 0;
PORTB = 0 ;

a = 150 ;
b = 250 ;

  while(1){
   if (PORTC.F0 == 0) PORTB = a ;
   if (PORTC.F1 == 0) PORTB = b ;
  }
}
```

【4】コンフィギュレーションビットの設定

プログラムが作成できましたので、続いてコンフィギュレーションビットの設定を行います。

コンフィギュレーションビットとは、HEXファイル(C言語で作ったファイルをコンパイルした後にできあがる、最終的にPICマイコンに書き込むためのファイル)をPICに書き込む時にのみ設定できるハードウエアの基本的な動作条件を設定する設定項目のことをいいます。
すごく重要な設定で、このコンフィギュレーションビットの設定が間違っていると、プログラムやハードウエアの設計が正しくてもプログラムが動かなかったり、期待した動作をしないなど・・・多くのトラブルの原因となります。

メニューバーの"Project"をクリックして"Edit Project"をクリックします。ダイアログが表示されます。
詳しい解説は後回しにして、とりあえず今回はデフォルト設定とします。
右下にある"Defailt"ボタンを押して下さい。

Defaultボタンを押してデフォルト設定にすると主な設定項目は下記のように設定されます。

・発振子の種類→HS	5MHz以上又はレゾネータ使用
・ウォッチドッグタイマー→OFF	ウォッチドッグタイマーの設定
・PowerUp Timer→OFF	電源投入時の起動遅延機能OFF
・MasterClear Enable→MCLR	MCLRピンはリセットピンに設定
・Brown Out Detect→Enabled	Vddが設定電圧より低下でリセット
・Low Voltage Program→Disabled	低電圧書き込み機能は無効
・BrownOut Reset Sel Bit→4.0V	Vddが4.0V以下で強制リセット

"OK"ボタンを押して元の画面に戻ります。

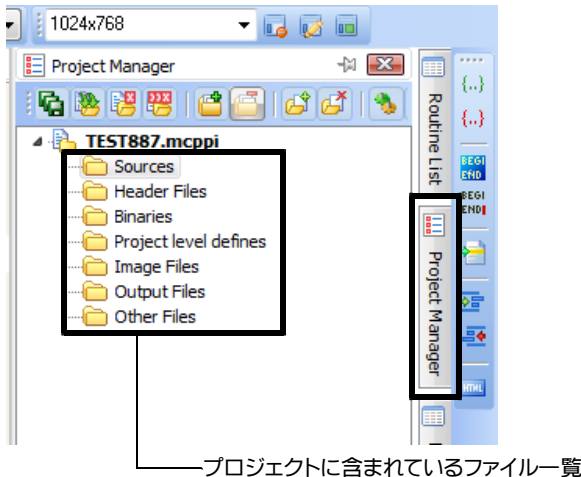
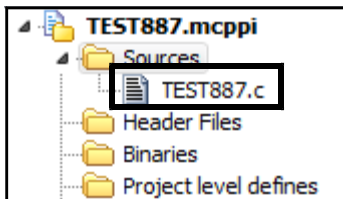
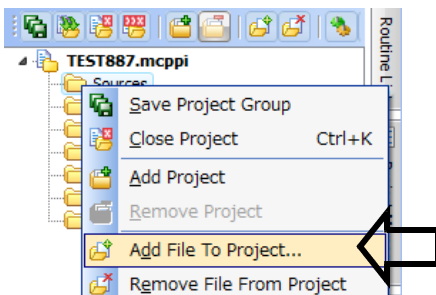
すごく大事なコンフィギュレーションビットの設定

PICの開発を始めると多くの方が戸惑うのがこのコンフィギュレーションビットの設定です。1つの原因は設定項目が英語だから。。もう1つはデバイス毎に設定項目が異なるからです。
プログラムが動かないと、「あれ?故障?もう故障!?」とすぐに機器やデバイスの故障を疑いたくなりますが、実際に機器やデバイスが故障することは滅多にありません。(使い方にもよりますが・・)
トラブルの最大の原因はコンフィギュレーションビットの設定が正しくないことです。コンフィギュレーションビットの設定がよく理解できていないと設定がおざなりになってしまい、デバイスを変えたりすると動かない・・・という原因になってしまうのです。
コンフィギュレーションビットの設定はちょっと複雑で難しいですが参考書やデータシートにも詳しく記載がありますので、ぜひそちらを読んでよく理解した上でご使用頂けることを願っております。

【5】プロジェクトへのファイルの登録

先にも説明しましたが、mikroCは、すべてのファイルをプロジェクトという単位で管理しています。よってプログラムを作成しても、そのソースファイル(拡張子.c)が、ファイルがプロジェクトに含まれていなければなりません。ビルドに先立って、必ず現在のプロジェクトにソースファイルが正しく登録されているか確認しましょう。

1 右サイドにあるサイドメニュータブの"Project Manager"をクリックします。現在作成しているプロジェクトに含まれる各ファイル類がツリー構造で表示されます。

プロジェクトに含まれているファイル一覧

2 "Sources"のツリーを展開します。ここに"TEST887.c"が含まれていればそのままで続行できます。
もし、何も登録されていない場合には、次の手順でソースプログラムを追加します。

"Sources"を右クリックするとメニューが表示されますので、その中から"Add File To Project"をクリックします。

ファイルを開くダイアログが表示されますので、先ほど作成したソースファイル "TEST887.c"を選択して開きます。すると、プロジェクトにソースファイルが追加されます。

※このプロジェクトに含まれるファイルを確認する作業はとても重要です。mikroCは、どんな作業をする場合であっても必ずプロジェクト単位で行うため、画面上に表示されている、されていないに関わらず、必ずプロジェクトに含まれているファイルに対して操作が行われます。よって、プロジェクトに正しくソースファイルが登録されているかどうか確認しておかないと、結果として正しいHEXファイルを作れずに、トラブルの原因となってしまうことがあります。

【6】ハードウエアの物理的な設定

PICD-500EX5ボードの各種設定を行いましょう。
今回はRBはLEDに接続し、RCは定常時プルアップに設定し、タクトスイッチが押された時にGNDに接続されるようアクティブLow設定にします。次のように設定してください。

・ディップスイッチSW6
→"PORTB"のスイッチのみON側にします。
・ジャンパーJ3(PORTC)
→"Pull-Up"間(上側)をジャンパーソケットでショートします。
・ディップスイッチSW3
→RC0とRC1のスイッチをON側にします。
・ジャンパーJ17(タクトスイッチ設定)
→"GND"側(下側)をジャンパーソケットでショートします。

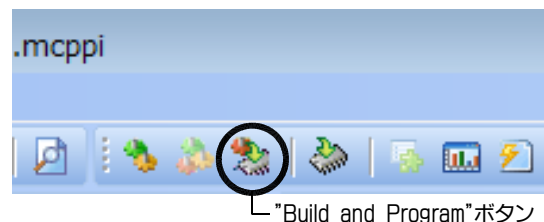
ボードの設定が正しくないとプログラムは期待通りに動作しません。
物理的な状態が作成したプログラムの内容に矛盾しないように正しく設定する必要があります。

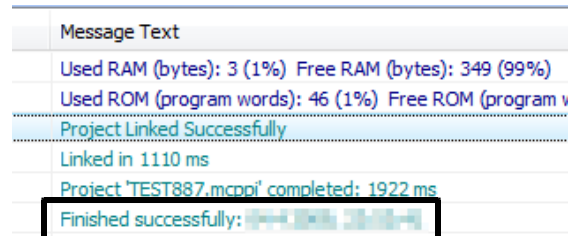
【7】ビルドと書き込みの実行

ビルドとは、C言語で記述したプログラムから、PICに書き込める形式のHEXファイルを作ることをいいます。PICにこのHEXファイルを書き込むと、PICが動作し始めるわけです。

mikroCと、PICD-500EX5の書き込み用ソフトウエアPICFlashは連動させて動作させることができます。ビルドでエラーがなければそのまま書き込みを実行し、プログラムの動作を実機で確認できます。

1 ツールバーの"Build and Program"ボタンをクリックします。

"Build and Program"ボタン

なお、キーボードからは"Ctrl"+"F11"キーを押しても実行できます。

2 メッセージウインドウに進捗状況が表示されます。
エラーが発生すると、赤文字で表示されます。エラーがない場合には、"Finished Successfully"と表示され、すぐに書き込みソフトウエアであるPICFlashが起動し、書き込みを開始します。

3 書き込みが終わると、書き込みソフトウエアはすぐに閉じて、プログラムがPICD-500EX5上で動きます。

※Build時にメッセージウインドウに"main function is not defined"とエラー表示された場合には、プロジェクトに正しくソースファイルが登録されていないことが原因ですので【5】からやり直して下さい。

【8】動作確認
書き込みが完了するとプログラムがPICD-500EX5ボード上ですぐに動きます。動作を確認してみましょう。

PICD-500EX5のタクトスイッチ群のなかにあるRC0のスイッチを押します。RBに接続されたLEDが点灯します。RC0なので変数aの値、ここでは150が2進数でLEDに現れます。

150は2進数表記では　10010110　となります。
RBのLEDは下記のように点灯しているはずです。

※○は消灯、●は点灯

同様にして、RE1のスイッチを押すと変数bの値、すなわち250が2進数としてRBのLEDに表示されます。

ここまでで「プロジェクトの作成」→「プログラムの作成」→「ビルド及び書き込みの実行」という一連の作業はできたことになります。

どんなプログラム作成でもこの一連の操作の流れはほぼ同じ場合が多いですので、よくご理解頂き次のチュートリアルもお試し下さい。

■プログラムの解説

void main(){…}
main関数です。C言語では1つのプログラム内に必ず1つのmain関数が必要です。voidはmain関数の型で戻り値がない関数であることを宣言しています。{ }で囲まれた範囲がmain関数の範囲となります。

unsigned int a , b ;
符号なしのint型変数を宣言しています。C言語では文の最後には必ず ; (セミコロン)をつけます。
符号なしのint型の場合扱える値は、0～65535までです。

TRISB = 0 ;
TRISC = 0x3;
TRISレジスタは、ピンの入出力方向を設定するレジスタです。
該当のビットが0の場合は出力、1の場合は入力となります。本チュートリアルではRBは全ピン出力、RCはRC0とRC1が入力なので0x3として設定しています。
C言語では16進数の値を表現する時は数値の前に0xを付けます。

PORTB = 0 ;
PORTBに0を代入しています。

ANSEL = 0;
ANSELH = 0;
PIC16F887では、PORTBはADコンバータのアナログ入力ピンとしてアサインされています。デジタルIOとしてピンを使用する場合にはANSELレジスタ及びANSELHレジスタに0を代入して、デジタルモードに設定します。
ANSELレジスタは、AN0～AN7まで、ANSELHレジスタはAN8～AN13までのピンのモードを決めます。該当ビットが0でデジタルIOに、1でアナログ入力ピンに設定されます。
ANSEL(H)レジスタは、電源投入時はすべてビットが1になっており、アナログ入力ピンに設定されています。よって、デジタルIOピンとして使用する場合には、プログラムの中で必ず両レジスタに0を代入してデジタルモードに設定しないと期待通りにプログラムが動かなくなってしまいます。

a = 150 ;
b = 250 ;
int型変数に値を代入しています。

while(1){…}
while文は( )内の値が1又は真であれば{ }内の処理を繰り返す繰り返し文です。ここでは評価式を1としていますので永久に{ }内が実行されます。
C言語では処理を永久ループさせたい場合にはよくwhile文を使用します。

if (PORTF.C0 == 0) PORTB = a ;
if文を使用した入力判定部分です。
( )内の評価式が真の時にその後ろに記述された文が実行されます。
mikroCでは「レジスタ名.Fビット数」でレジスタの1つのビットを指定できます。ここではRC0を指しています。
==(イコール2つ)は比較演算子の「等しい時」を表す演算子です。よくある間違いとして=を1つしか記述しない場合があります。=が1つの場合には代入演算子となってしまい正しく動作しません。IF文の時には==(イコール二つ)と覚えておくと間違いを減らせます。
PORTB = a　でPORTBレジスタに変数aの値を代入しています。その結果がLEDで表示されます。

チュートリアル② ～LCDの使用と、配列・ポインタ～

プロジェクトの新規作成やライブラリの登録、コンフィギュレーションビットの設定は、チュートリアル①と全く同じ方法ですので、チュートリアル①を参考にしてください。

新規にプロジェクトを作成して、下記のプログラムを作ってみましょう。

■プログラムの作成

PICD-500EX5に装着されている2行16文字のLCDに文字列を表示します。C言語では、文字列を扱う場合配列を使用します。
本チュートリアルでは固定した文字列をLCDに表示させてみましょう。

プログラムの作り方は、チュートリアル①と全く同じです。新しいプロジェクトを作成するとよいでしょう。コンフィギュレーションビットの設定は"Default"設定でかまいません。
次のようにプログラムを記述しましょう。

```
sbit LCD_RS at RB4_bit;
sbit LCD_EN at RB5_bit;
sbit LCD_D4 at RB0_bit;
sbit LCD_D5 at RB1_bit;
sbit LCD_D6 at RB2_bit;
sbit LCD_D7 at RB3_bit;

sbit LCD_RS_Direction at TRISB4_bit;
sbit LCD_EN_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;

void main() {
  char txt_a[] = "Hello";
  char txt_b[] = "World";

  ANSEL = 0;
  ANSELH = 0;

  Lcd_Init();
  Lcd_Cmd(_LCD_CLEAR);
  Lcd_Cmd(_LCD_CURSOR_OFF);

  Lcd_Out(1,1,txt_a);
  Lcd_Out(2,1,txt_b);
}
```

プログラムを記述した後はチュートリアル①と同様の手順で、ソースファイルがプロジェクトに正しく登録されていることを確認して、ビルドと書き込みを行ってみましょう。

LCDの1行目に"Microtechnica"と表示され、2行目に"PICD-500EX5"と表示されます。

■プログラムの解説

```
sbit LCD_RS at RB4_bit;
sbit LCD_EN at RB5_bit;
sbit LCD_D4 at RB0_bit;
sbit LCD_D5 at RB1_bit;
sbit LCD_D6 at RB2_bit;
sbit LCD_D7 at RB3_bit;

sbit LCD_RS_Direction at TRISB4_bit;
sbit LCD_EN_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;
```

main関数の前にsbitタイプを用いてLCDの物理的な配線状態を定義します。
sbitタイプは、PICマイコンのSFR(レジスター)の特定のビットに対してアクセスするため機能を提供します。
ここでは、その1つの使い方としてLCDの接続状態をあらかじめ定義しています。sbitは、TRISレジスタに対しても特別な書式として記述することができます。LCD関連のライブラリを使用する場合には、main関数の前で、この定義を記述してください。

※この定義をmain関数の中で記述してしまうとエラーになります。

char txt_a[] = "Hello";

char型配列を宣言しています。char型は文字を扱う8ビットの型です。C言語では文字はASCIIコードにて扱われます。
通常配列を宣言する際には要素数(インデックス値)を[ ]内に入力しますが、[ ]内を空欄にすると自動的に要素数が設定されます。
配列や変数を宣言する際に初期値を代入できますが、char型に限り文字列で初期化できます。"(ダブルクォーテーション)で囲むと文字列として認識されます。

Lcd_Init() ;

LCDを初期化する関数です。LCDを使用する前に記述します。

Lcd_Cmd(_LCD_CLEAR);

Lcd_Cmd( )関数は、LCDの制御コマンドを送信する関数です。
ここでは、画面をクリアーするコマンドと、カーソル(描画位置に点滅するカーソル)を表示しない設定にしています。
※mikroCのマニュアルは、"Help"→"Help"で見ることができます。

Lcd_Out(1,1,txt_a) ;

Lcd_Outは4ビットモードでLCDに文字や文字列を表示する時に使用します。書式は下記の通りです。

Lcd_(行, 列, char *text);

*textは、char型のポインタという意味です。ここでは、配列名を記述することで配列の0番目の要素のアドレス値を指定しています。

■配列とは?

配列は、連続したメモリー領域にスペースを確保する仕組みです。通常、変数を宣言するとメモリー上に格納スペースがコンパイラーによって割り当てられます。しかしこの作業はコンパイラーが自動的に行うものであり、開発者はどの変数がメモリー上の何番地に割り当てられたかからは分かりません。

文字列のように関係性が連続的なデータの場合、データは連続するアドレスに配置される必要があります。そこで変数ではなく配列を宣言します。配列を宣言するとコンパイラーは、連続したメモリー領域を確保します。配列は下記の書式により宣言できます。

```
型 配列名[要素数] ={初期値 0,初期値 1,・・・} ;
```

初期値は省略できます。char型配列の場合のみ文字列で初期化することができます。要素数を省略すると、自動的に要素数が決まります。

配列名を指定すると、その配列の要素数0のアドレス値が参照されます。要素数0のアドレスが分かれば、あとのデータは連続するアドレスに格納されていますので、配列のサイズさえ分かれば、データの連続性を損なわずにデータを取り出すことができます。

■ポインタとは?

ポインタは、通常の変数と異なり格納されている値を参照するのではなく、その変数に割り当てられているメモリー領域上のアドレスを参照する仕組みです。

例えば、 int a = 30; と宣言した場合、int型変数aは、初期値に30が代入された状態で、メモリー領域上のどこかに配置されます。しかし通常開発者は、変数aを参照しても30という代入された値は分かりますが、変数aがメモリー上の何番地に配置されているのかは分かりません。

ポインタ変数は、ある変数のアドレス値だけを専門に扱う変数です。C言語では、ポインタ変数を宣言する場合には、変数名の前に*(アスタリスク)を付けます。また、ある変数のアドレス値を取得したい場合には、変数の前に&演算子を付けます。下記に例を示します。

```
int a, b ;
int *p ;          }①

p = &a ;          ②

*p = 100 ;
b = *p + 1 ;      }③
```

①は、変数の宣言です。ポインタ変数pはint型変数のアドレス値を扱うためのものです。

②で、ポインタ変数pに変数aのアドレス値を代入しています。この時、ポインタ変数pには*が付かないことに注意します。

③ではポインタ変数pが指し示すアドレスにある変数、つまりは変数aに100を代入しています。ポインタ変数が指し示す先のアドレスの変数に値を代入する場合には、ポインタ変数に*を付けます。

さらに、ポインタ変数の指し示す先の変数、すなわち変数aの値に1を加算していますので、変数bの値は、101ということになります。

なぜこのように複雑な方法を使うのでしょうか。

ポインタ変数は、配列と組み合わせると色々な応用ができます。配列のように連続したアドレスに値が格納されている場合、ポインタで参照すれば、ポインタの値を増やしたり減らしたりするだけで、値を取り出すことができます。例えばシリアル通信で連続したデータを受信した場合、ポインタで先頭のアドレスを参照してあとはポインタの値をインクリメントするだけで連続したデータを参照できます。

チュートリアル③ ～7セグメントLEDの駆動とTMR0割り込み～

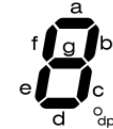
■プログラムの内容

PICD-500EX5に搭載の4桁の7セグメントLEDに数値0000～9999までを1秒間隔でカウントアップさせて表示するプログラムを作ります。7セグメントLEDは、ダイナミック点灯方式で点灯させる必要があります。コモンピンの駆動には時間制御が必要となりますので、本チュートリアルでは、TMR0の割り込みを使用して時間を作り出し、コモンピンを制御します。その他関数の作り方なども紹介します。

PICD-500EX5の7セグメントLEDは、各セグメントがPORTDに、コモンピンはトランジスタを介してRA0～RA3に接続されています。いずれも該当ピンがHレベルで点灯します。

■プログラムの仕組みを考えましょう

①4桁の数値を作る

要素数が4つの配列を作り、4桁の数値、すなわち0000～9999までの数値を桁ごとに配列に入れます。ただし、数値をそのまま入れても7セグメントLEDで数値を作れません。そこで、数値から7セグメントLEDのセグメントデータに変換する関数を作り、メイン関数から呼び出して7セグメントLEDの点灯させるセグメントデータに変換を配列に代入します。

例えば、3であれば、点灯させるセグメントは「a,b,g,c,d」です。PICD-500EX5では、7セグメントLEDの各セグメントにRDが接続されています。RDの値を0x4Fとすることで、数値の3が形成されます。

②各桁の値を取り出す

変数に4桁の値を入れますが、セグメントデータを作るためにそれぞれの桁の値を取り出さなければなりません。例えば、変数に1234という数値が入っている場合、千の位は1・百の位は2・・・と別々の数値として取り出します。数値の取り出しには除算と、あまりを返す算術演算子(%)を使用することにします。

例えば、変数iに数値1234が入っていて、百の位の数値を取り出すには work = (i / 100)%10 で取り出すことができます。

③TMR0割込を使って、コモンピンを時間制御する

PICD-500EX5の7セグメントLEDは、ダイナミック点灯にて制御します。TMR0はPICマイコンに内蔵された8ビットのタイマーで、256カウントをすることができます。カウンターが255になると、オーバーフローが起こり、割込が発生します。この割込を検出すれば、一定間隔で割り込みルーチンを実行できます。

割り込みルーチン先で、7セグメントLEDのコモンピンを制御して、各桁のセグメントデータを格納した配列の値を出力すれば数値が表示できます。

■プログラムの作成

```
char Ret_Work;
unsigned short disp_work, v, Common;
unsigned int i;
char disp_arr[4];

unsigned short Disp_Seg(unsigned short num) {
  switch (num) {
    case 0 : return 0x3F;
    case 1 : return 0x06;
    case 2 : return 0x5B;
    case 3 : return 0x4F;
    case 4 : return 0x66;
    case 5 : return 0x6D;
    case 6 : return 0x7D;
    case 7 : return 0x07;
    case 8 : return 0x7F;
    case 9 : return 0x6F;
  }
}

void interrupt() {
  PORTA = 0;
  PORTD = disp_arr[v];
  PORTA = Common;
  Common <<= 1;
  if (Common > 8u) Common = 1;

  v++ ;
  if (v > 3u) v = 0;
  TMR0   =   0;
  INTCON = 0x20;
}

void main() {
  OPTION_REG = 0x80;
  ANSEL      =   0;
  ANSELH     =   0;
  TMR0       =   0;
  INTCON     = 0xA0;
  v          =   0;
  Common     =   1;
  PORTA      =   0;
  TRISA      =   0;
  PORTD      =   0;
  TRISD      =   0;
  i          =   0 ;
```

↓次のページに続く

↓前のページからの続き

```
 while(1){
    disp_work = i / 1000u ;
    Ret_Work = Disp_Seg(disp_work) ;
    disp_arr[3] = Ret_Work ;

    disp_work = (i / 100u) % 10;
    Ret_Work = Disp_Seg(disp_work);
    disp_arr[2] = Ret_Work;

    disp_work = (i / 10u) % 10;
    Ret_Work = Disp_Seg(disp_work);
    disp_arr[1] = Ret_Work;

    disp_work = i % 10;
    Ret_Work = Disp_Seg(disp_work);
    disp_arr[0] = Ret_Work;

    delay_ms(1000);

    i++ ;
    if (i > 9999u)i = 0;
   }
}
```

■PICD-500EX5ボードの設定

このプログラムを正しく動作させるため、下記のようにPICD-500EX5ボードの各種設定をセットしてください。

・SW6はPORTA～Dはすべて0FF側に設定
・SW6のDIS3～DIS0はすべてON側に設定

■プログラムの解説

まず、プログラムはmain関数から動作を開始します。main関数では最初にレジスタの設定と、変数の初期化などを行います。

～main関数～

OPTION_REG = 0x80;

OPTIONレジスタの値をセットしています。本プログラムでは、TMR0のクロック源は内部クロックに、周波数を分周するためのプリスケーラーは、TMR0に使用、分周比は1:2に設定しています。

TMR0のオーバーフロー周波数は下記の式で計算できます。
「オーバーフロー周波数＝システムクロック÷4÷プリスケーラ分周比÷256」

システムが8MHzのシステムクロックで動作しているとして計算すると約3.9KHzとなります。逆数を取ると約256μ秒となり約256μ秒ごとに割込が発生することになります。
割込が発生すると、void interrupt()が実行されます。

INTCON = 0xA0;

INTCONレジスタは、各種割込の設定を行うレジスタです。
割込を有効に設定し、TMR0の割込を有効にしています。
TMR0は8ビットのカウンタで、0～255までカウントしますが、255から0に戻る際にオーバーフローが発生します。オーバーフローが発生するとINTCONレジスタのビット2が1になります。これにより割込の発生をソフトウエア側で知ることができます。
なお、このINTCONレジスタのビット2は、ソフトウエア側で手動で0に戻す必要があります。

disp_work = i / 1000u;
変数iの値を1000で割っていますが、C言語では数値の後ろにuを付けると符号なし整数として認識されます。

Ret_Work = Disp_Seg(disp_work);
Disp_Seg関数を呼び出しています。引数としてdisp_work変数の値を渡しています。これを値渡しと呼びます。渡せる値はunsinged short型のみです。
Disp_Seg関数からの戻り値は、Ret_Work変数に格納されます。

disp_work = i / 1000u ;
Ret_Work = Disp_Seg(disp_work) ;
disp_arr[3] = Ret_Work ;
一連の動作をみてみましょう。例えば i = 5000 とした場合・・・
5000÷1000となり千の位の値5がdisp_work変数に入ります。
その値5は、Disp_Seg関数に引数として値が渡されます。
Disp_Seg関数では、switch文によって処理を分岐し、数値5に対応する値、0x6Dを戻り値として返します。
0x6Dは、変数Ret_Workに代入されます。その値は、7セグメントLEDの各桁の値を格納する配列の要素3に入ります。

i++;
変数iの値をインクリメント(1増加)させています。

～Disp_Seg関数～
unsigned short Disp_Seg(unsigned short num) {
関数を宣言しています。関数を作る書式は下記の通りです。

型　関数名(型 引数1, 型 引数2, ・・・) { }

この関数は、unsigned short型の値を引数として受け取ることができます。またこの関数は、unsigned short型の値を戻り値として戻します。引数はいくつでも作ることができ、引数がない場合には( )だけにします。また戻り値がない関数の場合にはvoid型とします。
関数からは、return文によってmain関数に戻れます。

switch (num) {
switch文は、( )内の評価式を評価してcaseで指定された値のルーチンに処理を分岐します。ここでは、引数として受け取った値を格納してあるnumの値を条件式としています。

～割込関数～
void interrupt() {
mikroCでは、割込が発生すると必ずこの関数が呼び出されます。main関数実行中でも、割込発生で直ちにこちらの関数が呼び出されます。

Common <<= 1;
<<は左シフトの演算子です。ここでは、Common変数の値を1ビット左シフトしてその値をCommon変数に代入しています。1ビット左シフトすることで、RAの出力が1ビットずつ左方向にシフトします。

TMR0 = 0;
INTCON = 0x20;
TMR0を0に戻し、INTCONレジスタの値を再度0x20に初期化しています。INTCONレジスタはTMR0の割込発生後、オーバーフローを通知するビット2が1となり、このビットはソフトウエア側でリセットする必要があります。

チュートリアル③　～非同期式シリアル通信とADコンバーター～

■プログラムの内容
非同期式シリアル通信(UART)を使用して、RS232C通信をするプログラムを作ります。また、ADコンバーター及び、変数の値をLCDに表示する方法も紹介します。
ADコンバーターのチャンネル2(AN2=RA2)からアナログ電圧値を読み込み、その値をLCDに表示します。また、UART経由で文字"A"を受信すると、ADコンバーターの値をUARTで文字列として出力します。

■プログラムの作成

```
sbit LCD_RS at RB4_bit;
sbit LCD_EN at RB5_bit;
sbit LCD_D4 at RB0_bit;
sbit LCD_D5 at RB1_bit;
sbit LCD_D6 at RB2_bit;
sbit LCD_D7 at RB3_bit;

sbit LCD_RS_Direction at TRISB4_bit;
sbit LCD_EN_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;

void Prog_init();

void main() {
  unsigned char UART_Res;
  unsigned int AD_Res;
  char disp[7];
  unsigned short i;

  Prog_init();

  while (1) {
    AD_Res = ADC_Read(2);
    IntToStr(AD_Res,disp);
    LCD_Out(1,1,disp);
    if (UART1_Data_Ready()) {
       UART_Res = UART1_Read();
       if (UART_Res == 'A'){
          for (i=2; i<6; i++){
            UART1_Write(disp[i]);
          }
       }
    }
  }
}
```

↓次のページに続きます。

```
void Prog_init() {
  ANSEL = 0x04;
  ANSELH = 0;
  ADCON0 = 0xC9;
  ADCON1 = 0x80;
  TRISA = 0xFF;
  TRISD = 0;
  TRISC = 0;
  LCD_Init();
  LCD_Cmd(_LCD_CURSOR_OFF);
  LCD_Cmd(_LCD_CLEAR);
  UART1_init(9600);

  return;
}
```

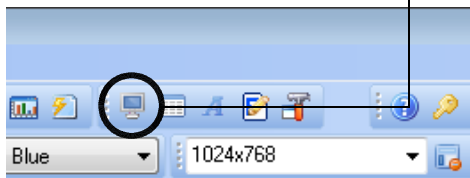
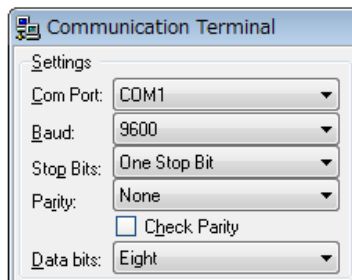
■PICD-500EX5ボードの設定

このプログラムを正しく動作させるため、下記のように設定します。

・SW6及びSW9の1番～6番まですべてOFF側に設定
・SW7はRC7をON側に設定
・SW8はRC6をON側に設定
・J16は"RA2"と書かれた部分をソケットでショート
・LCDが装着されていることを確認します

■パソコン側の準備

このプログラムでは、RS232C通信を行います。パソコンのRS232Cポートと、PICD-500EX5の"RS232"とかかれたポートを全結線ストレートケーブルで接続してください。

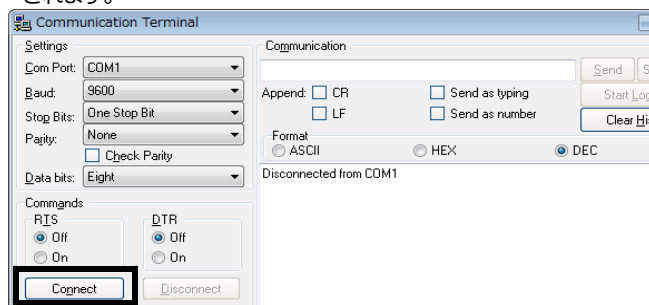
RS232Cのターミナルソフトはmikro Cに付属しています。このターミナルソフトを使用すると大変便利に通信が行えます。
mikroCのツールバーにある"USART Terminal"のアイコンをクリックします。

ターミナルソフトが起動します。
"Settings"グループにて通信に関する基本的な設定を行います。

今回は、通信速度9600bps、1ストップビット、ノンパリティーの設定にしますので、上記のように設定してください。
"Com Port"は、PICD-500EX5と接続したRS232Cのポート番号を設定してください。その他の設定はデフォルト設定にしておきます。

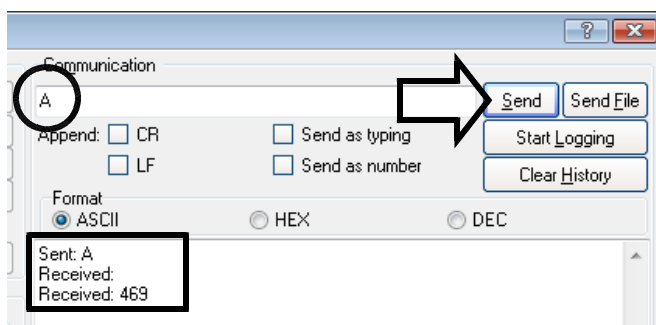
■コンパイルと書き込みの実行

これまでのチュートリアルと同様、ビルドしてHEXファイルをPICD-500EX5のマイコンに書き込んでください。

■動作確認

①まず、ADコンバーターの値がLCDに表示されることを確認します。
　P2の可変抵抗器を回転させると、LCDの表示が0～1024の間で表示されます。

②次にRS232CでADコンバーターの値を取得してみましょう。
　ターミナルソフトの"Connect"ボタンを押すとRS232C通信が開始されます。

　今回は受信するデータは文字列ですので"Format"の部分は、"ASCII"を選択します。

③ターミナルソフトの"Communication"のテキストボックスに大文字でAと入力して"Send"ボタンをクリックします。
　受信ウインドウに Sent:A と表示され次の行に"Received: xxx"と数値が表示されれば成功です。

　LCDに表示されている数値と同じ数値がRS232C経由でも受信できることを確認します。また、P2のボリュームを回転させると、ADコンバーターの値が変わりますので、戻り値が変わるのが確認できます。

■プログラムの解説

void Prog_init();
　今回のプログラムでは、レジスタの初期化と各機能の初期化をmain関数内ではなく、新しくPro_initという名前の関数を作って、そこで行っています。
　関数を自分で作る場合には、main関数の外（プログラムの先頭）で、自作した関数を宣言しておきます。
　voidは戻り値の無いvoid型関数を作ることを意味しています。

ANSEL = 0x04;
ANSELH = 0;
ADCON0 = 0xC9;
ADCON1 = 0x80;

ADコンバータ関連のレジスタの設定です。
ANSELレジスタはAN0～AN7のどのチャンネルをアナログ入力にするかを決めるレジスタです。今回はAN2をアナログ入力にしますので、ANSELレジスタに0x04を代入しています。（該当ビットが1の時アナログ入力ピンとなります。）

ADCON0レジスタは、ADコンバータ用のクロック源の選択及びチャンネル選択、ADコンバータ機能の有効／無効設定のレジスタです。今回はADコンバータ用に搭載されている内蔵RC発振子をクロック源として選択しました。また、チャンネルはAN2を選択しました。
ADコンバータの機能を有効に設定しました。
ADCON1レジスタは、ADコンバータからの10ビットの戻り値を左詰めにするか、右詰めにするかの設定と、基準電圧の設定を行います。mikroCではADコンバータの結果は右詰めとします。
基準電圧は、VssとVddに設定しています。（PICの電源電圧の0V～5Vが基準電圧となります）

Uart1_Init(9600);

UARTの初期化を行う関数です。( )内には通信速度をbps単位で記述します。ここでは9600bpsに設定しています。

AD_Res = Adc_Read(2);

Adc_Read()関数は、( )内に指定したチャンネルからAD変換の値をunsigned型で取得し、変数に代入します。PIC16F887に搭載のADコンバーターは10ビットの分解能がありますので、unsigned int型の変数に値を代入します。

IntToStr(AD_Res,disp);

型変換です。C言語では型の扱いが厳密であるため、変数や関数を使用する場合には常に型を意識しなければなりません。
先の命令でADコンバーターで取得した値は、符号なしのINT型変数に代入しました。これは、Adc_Read関数が返す値がINT型だからです。
一方Lcd_Out関数で出力する文字列を扱う場合には、必ずchar型ポインタ変数でなくてはなりません。char型のポインタ変数は、char型配列の先頭アドレスです。mikroCでは、各変数の型からstring（文字列形式）にする関数を持っています。
IntToStr関数は、INT型変数の値を文字列形式に変換します。文字列形式とは、すなわちchar型の配列です。char型のdisp_txt配列を宣言して、そこに値を代入しています。なお、配列の大きさは型変換関数によって定められており、IntToStr関数では要素数は7にすることが決まっています。

LCD_Out(1,1,disp_txt);

型変換によって変換されたAD_Res変数の値をLCDに出力しています。なお、IntToStr関数では7キャラクター分を出力しますので、LCDでは数値の前部分にブランクが入ります。

※7キャラクタ分ですが文字列の最後は¥0が入ります。

if (Uart1_Data_Ready()) {

Usart_Data_Ready()関数はUARTの受信バッファにデータが格納されると真になります。これをif文で検出することでデータの受信を監視します。

UART_Res = Uart1_Read()

char型の変数recに受信データを取り込みます。1バイトサイズのデータのみ受信できます。連続するデータの場合には、配列とポインタを使用します。

for (i=2; i<6; i++){

所定の回数繰り返し処理を実行するFor文です。
ここでは、変数iの値の初期値を2として、iの値が5以下の時、処理を実行します。i++は、処理が1回実行されるごとにiの値をインクリメントする記述です。
IntToStr関数によって、ADコンバーターの値はサイズ7の配列に文字列として格納されています。それを1要素ずつUARTで出力するため、要素2～要素5までの4桁をこのFor文で出力します。

Uart1_Write(disp[i]);

Usart_Write()関数は、UART経由で( )内のデータを出力します。
配列disp_txtの内容を、For文で繰り返しながらUART経由で出力します。

■グローバル変数とローカル変数

変数や配列はプログラム内で使用する場合には、あらかじめ型を指定して宣言しなければ使用できません。
この時、宣言する"場所"によって、グローバル変数とローカル変数に分けることができます。

グローバル変数は、main関数やその他の関数の外で宣言した変数や配列です。先の7セグメントLEDの例では、プログラムの先頭で宣言しており、これらはグローバル変数となります。
グローバル変数とは、すべての関数から共通して値を参照したり、代入することのできる変数、配列です。異なった関数から共通の変数を使用できますので、ある関数で行った演算結果をグローバル変数に代入し、その値をまた別の関数で参照する・・・といった使い方ができます。

それに対してローカル変数とは関数内で宣言した変数のことで、その変数はその関数内でしか参照できません。他の関数からは値を参照したり代入したりすることができない変数のことをいいます。

宣言をする"場所"によってグローバル変数とローカル変数に分かれますので、プログラムの内容に応じて使い分けることができます。

チュートリアル④　～ICD機能を使ったデバッグを体験する～

■ICD機能とは?

ICDとはインサーキットデバッグの略で、ターゲットボード上のデバイスに実際にプログラムを書き込み、動作させながらパソコン上でデバッグを行う手法のことです。
例えば現在実行している行をハイライト表示させながら実機での動作が見られますので、プログラムの挙動を簡単に把握できます。また、1行ずつプログラムを実行したり、変数やレジスタの現在の値をリアルタイムに閲覧することができたりします。
本セットに付属のmikroCには、C言語レベルでデバッグのできるICD機能が搭載されています。

■ICD機能を使うには

ICD機能を使用する場合、プログラムには特に大きな変更点はありませんが、delay_(ms)などのdelay関数が入ったプログラムは正しく動作しません。ICDを使用する場合には、delay関数をプログラム中からなくして実行してください。また、関数によっては、1つの関数で大量の処理を実行するため、ステップ実行(1行ずつ実行)では、時間がかかりすぎる場合があります。

ICD機能を使用する場合、コンパイル時に"Build type"を"ICD debug"設定にする必要があります。
このチュートリアルを通してICDの使い方を一通り体験してみましょう。

■プログラムの内容

LCDに文字"microtechnica"を表示されます。
但しICD機能の動作を確認するため1文字ずつ表示するプログラムを作ります。

■プログラムの作成

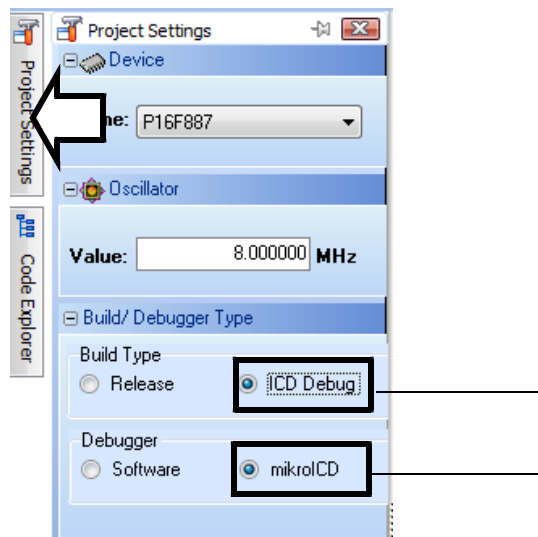
新しいプロジェクトを作成して、プログラムを作ってください。
プロジェクトの作成方法等は従来の方法と全く違いはありません。

```
sbit LCD_RS at RB4_bit;
sbit LCD_EN at RB5_bit;
sbit LCD_D4 at RB0_bit;
sbit LCD_D5 at RB1_bit;
sbit LCD_D6 at RB2_bit;
sbit LCD_D7 at RB3_bit;
sbit LCD_RS_Direction at TRISB4_bit;
sbit LCD_EN_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;

char text[14]="microtechnica";
int i;
```

↓次のページに続きます

```
void main(){
     ANSEL = 0;
     ANSELH = 0;
     TRISB = 0;
     Lcd_Init();
     Lcd_Cmd(_LCD_FIRST_ROW);
     Lcd_Cmd(_LCD_CURSOR_OFF);
     Lcd_Cmd(_LCD_CLEAR);

     for(i=0 ; i<13 ; i++){
       Lcd_Chr_Cp(text[i]);
     }
}
```

■コンパイルと書き込み

プログラムが書き終わったら、コンパイルをします。このとき、従来とは違い、Build typeをICD用に変更します。
画面左側のサイドメニュータブの"Project Settings"をクリックしてメニューを展開します。"Build type"のチェックボックスのうち、"ICD debug"と"mikroICD"にチェックを入れます。

チェックを入れたら、ビルドして、従来どおりPICマイコンへ書き込みを行ってください。
"ICD debug"オプションを指定してコンパイルしたプログラムは、PICマイコンへ書き込んだ後にすぐには動作しません。

■ICDを動作させてみる

ICDを実行して、プログラムの動作内容を確認しましょう。
LCDに何も文字列が表示が出ていないことを確認します。文字列が表示されている場合には一度PICD-500EX5の電源を切断してLCDの画面をクリアした状態で下記を実行してください。

1　メニューバーの"Run"→"Start Debugger"をクリックします。
　キーボードの"F9"キーでも同様の操作ができます。

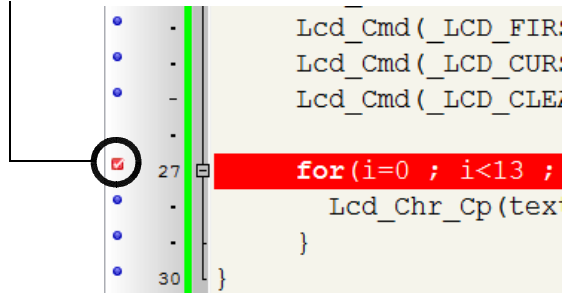
2　ANSEL = 0;　の行が青色バーでハイライトされます。ICDでは実行される行が青色バーでハイライト表示されます。
　ここでは、ステップ実行をしますがLCD関連の関数は処理が多いため、ステップ実行をすると処理に時間がかかってしまいます。よって、for文の所にブレークポイントを設定してそこまでは通常モード

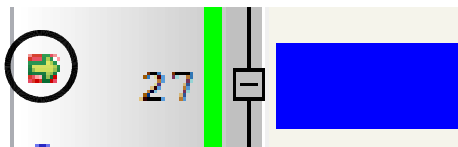
でプログラムを実行させて、for文以降からステップ実行で実行して動作を確認しましょう。

※ANSEL=0; の行が青くハイライトされず、別のアセンブラの画面が表示されてしまった場合には、メニューバーの"Run"→"Disassembly mode"をクリックして、画面をC言語の画面に戻してください。

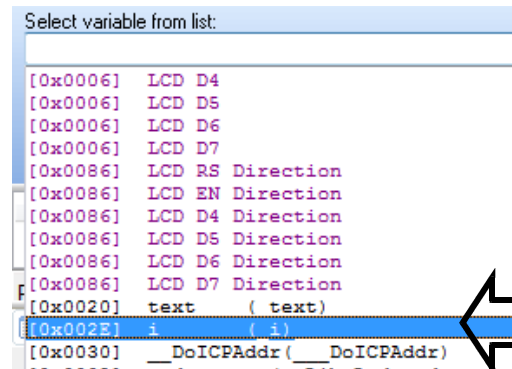
※ブレークポイントとは、ICDを実行した時プログラムの実行をストップさせたいポイント(行)のことを言います。

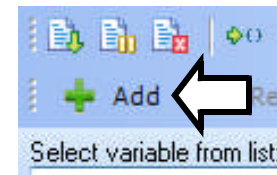
3 for文の行にブレークポイントを設定します。for文の記述がある行の左側に●印が表示されています。
この●をクリックすると、●印が"レ"印に変わり、指定した行が赤色でハイライトされます。

ここの●印の部分をクリック

4 ブレークポイントまで一気にプログラムを実行させてしまいますので、キーボードの"F6"キーを押します。
※メニューバーの"Run"→"Run/Pause Debugger"をクリックしても同じです。
→for文の一行が青色バーでハイライト表示されます。

5 LCDにはまだ文字は表示されていないはずです。ブレークポイントを解除します。矢印とレ印が重なっている部分を1回クリックします。

6 この状態で1行ずつ実行するステップ実行を行い、1文字ずつLCDに文字が表示されるのを確認しましょう。
キーボードのF8キーを押します。又はメニューバーの"Run"→"Step Over"をクリックします。F8キーを押すと、青色バーでハイライト表示された部分が移動します。

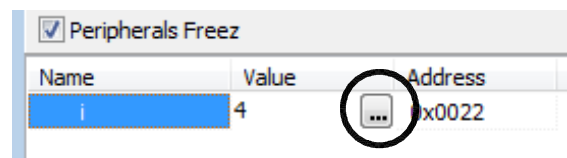
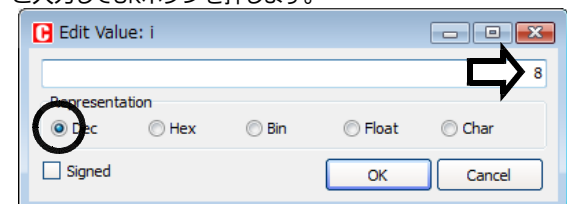
7 F8キーを2回押すと、LCDに"m"と表示されます。
同様にして、F8キーを何回も押すと、プログラムはFor文を繰り返し実行して、1文字ずつLCDに文字が表示されていきます。5文字程度表示されたら次の手順を実行します。

8 ICDを実行すると、ウインドウの右側にWatch Windowが表示されます。ウォッチウインドウは、変数やレジスタの値をリアルタイムで閲覧できるウインドウです。ここでは変数iの値が1ずつ増えていく様子を確認しましょう。
変数iを追加します。"Select Variable form list"のプルダウンをクリックして、その中から
「[0x---] i (_i)」と表示されている部分クリックします。

通常はリストの一番下の方にあります。

9 Addボタンを押して、追加します。なお、リストから見つけにくい場合には、検索することもできます。
その場合には、"Search for variable by assembly name"の部分に _i と入力します。変数名は、_(アンダーバー)に続けて変数名を入力します。

10 リストに追加されるとNameとValue、Addressが表示されます。Valueの値は現在の変数iに格納されている数値です。F8キーを押してFor文を実行すると、1ずつiの値が増えていく様子を見ることができます。

値が更新されると数値は赤文字で表示されます。

11 この値はユーザーが自由に変更することができます。
変数iは、text配列のインデックス値となっていますので、変数iの値を変えるということは、表示する文字を変えると言うことになります。ここでは試しに変数iの値を8にしてみましょう。

変数iのValueの値をクリックします。すると、横に四角いボタンが表示されますのでクリックします。

12 Edit Value: i というダイアログが表示されますので、"Representationの部分から10進数を意味する"Dec"をクリックします。値を8と入力してOKボタンを押します。

13　変数iの値を強制的に8に変えましたので表示される文字列も変わるはずです。キーボードのF8キーを押して実行してみましょう。文字が変わることを確認します。
※8文字目は、hなのでLCDには、hが表示されます。

ICD機能を終了する場合にはメニューバーの"Run"→"Stop Debugger"をクリックします。

■ICD機能を使用しない場合の注意

ICD機能を使用する場合、コンパイル時のBuild typeをICD用に設定しました。このICD用に設定されて生成されたHEXファイルは、通常動作はしません。

ICD機能を使用しない通常のHEXファイルをビルドしたい場合には必ずビルドする前に"Build type"から"Release"にチェックを入れて、ビルドを行ってください。

mikroCのサンプルプログラムについて

mikroCの各種サンプルプログラムは、PIC16F887用とPIC16F877A用のものが、CD-ROMに収録されています。

CD-ROMの"Cコンパイラー"フォルダ内の"サンプルプログラム"フォルダに収録されております。上記ディレクトリより対応PICを選択してご使用ください。

なお使用時は、CD-ROMからハードディスクにコピーしてお使いください。CD-ROM内からは直接使用できません。

mikroCのサポートフォーラム

mikroCは、世界中で使用されているユーザーの大変多いCコンパイラーです。mikroCの開発元であるmikroElektronika社では、サポートフォーラムを開設しており、活発な意見交換が行われています。

言語は英語ですが、検索をすることで色々な手法や問題の解決策が掲載されていますので、ぜひご利用ください。

http://www.mikroe.com/forum/

なお、記事の閲覧だけであれば登録は必要ありませんが、投稿する場合にはメンバー登録が必要になります。

アセンブラ言語のチュートリアルについて

MPLABを統合開発環境して用いて、アセンブラ言語にてプログラムを開発する方法について記載したチュートリアルは、付属のCD-ROMに収録されている電子マニュアルに記載しております。